*Intelligent Information Management*, **2012, 4, 338-347**
http://dx.doi.org/10.4236/iim.2012.46038 Published Online November 2012 (http://www.SciRP.org/journal/iim)

# Intelligent Multi-Agent Based Information Management Methods to Direct Complex Industrial Systems

**Danilo Avola[1], Luigi Cinque[2], Giuseppe Placidi[1]**
[1]Department of Life, Health and Environmental Sciences, University of L'Aquila, L'Aquila, Italy
[2]Department of Computer Science, Sapienza University of Rome, Rome, Italy
Email: danilo.avola@univaq.it, cinque@di.uniroma1.it, giuseppe.placidi@univaq.it

Received September 1, 2012; revised October 1, 2012; accepted October 15, 2012

## ABSTRACT

In recent years, the increasingly complexity of the logistic and technical aspects of the novel manufacturing environments, as well as the need to increase the performance and safety characteristics of the related cooperation, coordination and control mechanisms is encouraging the development of new information management strategies to direct and manage the automated systems involved in the manufacturing processes. The Computational Intelligent (CI) approaches seem to provide an effective support to the challenges posed by the next generation industrial systems. In particular, the Intelligent Agents (IAs) and the Multi-Agent Systems (MASs) paradigms seem to provide the best suitable solutions. Autonomy, flexibility and adaptability of the agent-based technology are the key points to manage both automated and information processes of any industrial system. The paper describes the main features of the IAs and MASs and how their technology can be adapted to support the current and next generation advanced industrial systems. Moreover, a study of how a MAS is utilized within a productive process is depicted.

**Keywords:** Industrial Systems; Information Management; Intelligent Agents; Multi-Agent Systems

## 1. Introduction

The current industrial systems are destined to become environments even more technologically advanced in every aspect. This is due to several reasons: the complexity of the new manufacturing processes, the need to optimize costs, the increasing of the complexity in managing resources, the need to improve performances, the aspects tied to the information exchanges, the manufacture of complex products, the need to ensure advanced safety standard, and so on. All these aspects imply a continuous and growing renovation of the current industrial activities. Artificial Intelligence (AI) techniques [1,2], utilized to overcome these new qualitative and quantitative challenges bring from the next industrial age, seem to provide the more suitable support since they naturally have all these technical features (e.g., adaptability, reasoning, learning) required from the new manufacturing processes [3]. In particular, the use of AI based strategies allow to the industrial environments to adopt autonomous, dynamic and intelligent procedures to face expected and unexpected issues. The IAs and MASs are technologies that can play an important role in every aspect involved in the development of the next generation of the advanced industrial systems. In fact, these systems have to be designed to include intelligent, autonomous and evolvable entities in turn composed by different sub-entities having the same features. The number of levels (*i.e.*, entities, sub-entities) and the complexity of each entity and sub-entity depend on the specific kind of the industrial system. The mentioned features are strongly tied to the IAs and MASs paradigms [4,5]. Moreover, several IAs and MASs have been just adopted in a wide range of applications that are connected to advanced industrial systems, such as: Robotic, artificial vision, production planning, systems control, engineering, information exchanging, and so on. In what follows, we describe the main features of the IAs and MASs and how their technology can be adapted to support the current and next generation advanced industrial systems.

The paper is organized as follows. Section 2 introduces the general aspects of the logical architecture of the current industrial systems. Section 3 describes the main features of the IAs and MASs to be adapted to support industrial systems. Section 4 provides a study of how a MAS is utilized within a productive process. In particular, the section discusses the different MASs algorithms highlighting their use to support the real-time production process. Section 5 shows the principles of the hybrid systems and presents the main characteristics of a case study. Section 6 concludes the paper.

 

## 2. Current Industrial Systems Architecture

The logical architectures of the current industrial systems can be seen as a correlated set of connected crucial processes that have to achieve a prefixed objective taking into account the current elaboration state of every other process. All the processes are aimed to reach a common target: The Artifact. In our context, the last mentioned term has to be considered both a physical object (e.g., technical or mechanical device, industrial product) and an abstract object (e.g., general service, immaterial product). The concepts regarding the creation or delivery of an artifact can be generalized to provide a common description of current industrial system architectures.

Each process deals with a specific activity (or part of it) of the whole industrial process. In fact, a generic Industrial system includes heterogeneous activities, such as: economical planning, quality and test control, real-time production control, internal activities, processes monitoring, robot control, sales planning, and so on. For this reason, the specification of a process can be completely different from another. As shown in **Figure 1**, a process is an autonomous entity characterized by internal rules and protocols. Every process works in coordination with others by taking into account their internal state.

The internal rules and protocols are highly dependent from the specific process. In general, the output of a process (*i.e.*, artifact and information) regards a collection of homogeneous activities, while the input (*i.e.*, processes interaction) concerns the connection of several heterogeneous activities, such as: Technical and schedulling information, control and manufacturing activities, data, artifact (or part of it). Every single process is composed of several different tasks (or sub-tasks) concerning a specific part of the whole industrial process. As shown in **Figure 2**, a task (or a sub-task) is an activity that can be solved, by decisional processes, using autonomous and intelligent methods without a specific knowledge of the surrounding processes. The depth of the tree depends

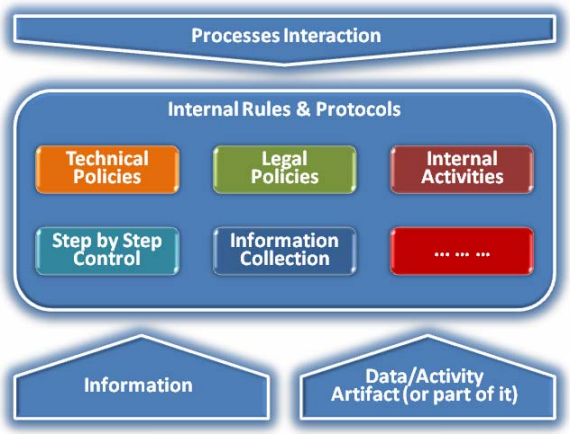


**Figure 1. A general scheme of a process.**

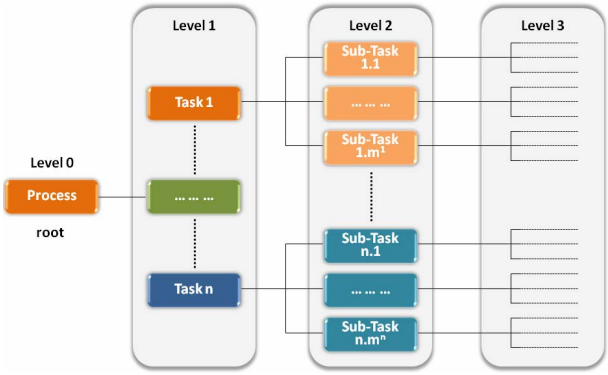


**Figure 2. Process, tasks and sub-tasks hierarchy.**

on both the kind of process and its complexity. Usually, processes concerning long term economic planning and technical activities are more complex than others.

The process at the root level (*i.e.*, level 0) provides each of its tasks the needed input to accomplish the related assignments. Likewise, each further task (*i.e.*, level 1) provides each of its sub-tasks the suitable part of the original input to accomplish the related sub-assignments. This activity of splitting input and assignments ends at the last level of the tree. The solution of each task and sub-task, by a decisional process, can require information or activities coming from different tasks and/or sub-tasks connected to any tree level. For this reason, the interaction mechanisms are a critical aspect of the current industrial systems. A failure in a task can produce a critical crash of the related process, or of the connected processes. Furthermore, the last mentioned aspect is fundamental to establish the intelligent mechanisms to solve the heterogeneous activities belonging to the complex industrial systems (e.g., real-time monitoring, dynamic business planning, chain control, simulation environments) and to overcome the expected and unexpected issues (e.g., breakdowns, environment changes, damages).

The decisional process that solves a task at the level 1 of the hierarchy can interact with any other decisional process that solves any other task at the same hierarchical level. Subsequently, the decisional processes that solve sub-tasks on the others levels of the hierarchy (e.g., level 2) have to directly interact only with the decisional processes that solve a sub-task connected at the same task or sub-task at the previous hierarchical level.

In others worlds, on each tree level, only the "brothers" can directly interact. If decisional processes belonging to different "fathers" have to interact, then the same "fathers" will accomplish the interaction activity.

As shown in **Figure 3**, the artifact is the result of joined processes. Likewise the previously described cooperation strategy, each decisional process that solves a process within the level 0 can interact with any other

 

decisional process that solves any other process at the same level 0. Also in this case, if two decisional processes, driving two different tasks (e.g., at level 1) belonging to two different processes, have to interact, the same two processes will accomplish the interaction activity. This kind of hierarchical interaction serves to avoid disorder (*i.e.*, anarchy) between tasks, sub-tasks and processes. Moreover, it is useful to allow the decisional process to have a total autonomy with respect the fixed assignment.

Each decisional process involved in the solution of a process, task or sub-task has to have well defined features. These features specify the way (e.g., protocols of communication and execution, priority of the decisional tree, priority of the queues, relationships between tasks or sub-tasks) and the resources (e.g., technical, economical, time planning, supply chain management) to carry on the activities. A generic decisional process can be considered as a set of activities that can be carried on by an intelligent, autonomous and flexible entity. Technically, these entities can be seen as a set of interactive procedures able to learn and evolve according to their life cycle. For these reasons, the use of IA and MAS technologies are considered the best solutions to these kinds of issues. As shown in **Figure 4**, each decisional process should be driven by a single IA. Each IA knows only the rules, the procedures and the protocols related to its specific task. It has a limited point of view regarding the whole industrial process and it is able to adopt solutions only on its specific contextual environment. Moreover, the networked set of the IAs (*i.e.*, MASs) should drive each process into the industrial system. This association between tasks, assignments and intelligent entities allows to implement independent customized solutions to solve global and local specific issues.

## 3. IAs and MASs in Industrial Systems

The agent and multi-agent technologies have been widely adopted in robotic and automatic control system fields. The growing complexity of the current industrial environments and their affinity to the mentioned fields has encouraged the design of IAs and MASs aimed to direct

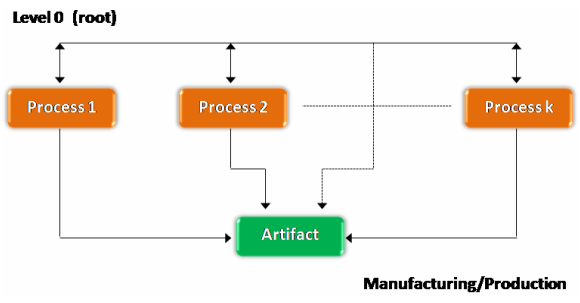


**Figure 3. Level 0: The processes.**

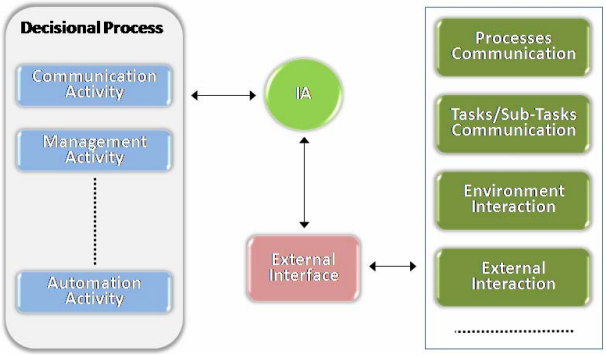


**Figure 4. Decisional process and intelligent agent.**

each aspect of the industrial chain. The CI based techniques promise to be the best way to perform these historical changes. There are several similar definitions of intelligent agent [6,7]. Considering the set of these definitions, in our context an intelligent agent can be described as: A social, autonomous and intelligent computational entity capable to accomplish a definite objective through an adaptive reasoning. Regardless of how an IA is defined, it is characterized by several general features [8] that can be adapted to our context. These features, joined with the given definition, complete the IA paradigm.

### 3.1. IA Properties Adapted to Industrial Systems

There are five main properties belonging to the IA that have to be adapted to build a basic industrial oriented IA.

#### 3.1.1. Reactivity

An IA is a reactive entity that continuously perceives and affects with its surrounding environment. In an industrial system this feature depends on the process on which the agent is applied. For example, in economical planning processes the reactivity can be considered a not active feature, *i.e.*, the agent has a starting receptive state and it will tend to switch its condition only if it recognizes particular changes in the environment. Instead, in technical processes the reactivity can be considered active, *i.e.*, each change in the environment is considered an important event that has to be analyzed in real-time.

#### 3.1.2. Pro-Activity

An IA has to operate in an expected way, but it can also take initiatives to overcome unexpected events. Usually, this feature does not depend on the process, it is tied to the environment in which the agent has to operate. The environments into the industrial processes can be classified in deterministic and non-deterministic. At the first class often belong processes derived from economical tasks that are tied to several unexpected changes (e.g., economical plans, supply chain management). At the last

 

class almost always belong processes derived from technical tasks that are usually scheduled and coordinated (e.g., test and control, robot interaction, real-time monitoring).

### 3.1.3. Autonomy

An IA has to be capable of autonomous actions to accomplish its objectives. There are several situations in which an agent has to autonomously decide what is necessary to do, and how it has to be done. The main issue of this feature is to decide which, how many and when specified actions/activities can be adopted by the IA. As general rule, an IA is authorized to take autonomous decisions only if there is the real possibility that the whole process crashes. These situations occur quite often in advanced industrial systems where several technological actors are involved. In fact, the introduction of complex systems in the industrial environments (e.g., real-time systems, on-line robot vision monitoring, automatic control management) has led high level of entropy. Usually, a reasoning decision tree is used to decide the actions that have to be performed [9,10]. By the autonomy property each agent can interact with the environments or other agents without an external presence to ensure the achievement of the prefixed objectives.

### 3.1.4. Flexibility

An IA has to interact with the surrounding environment in different ways. It has to have the capability to adopt quickly itself to drastic changes into the relationships, environments or events. Sometimes conventional communication or interaction ways could be not sufficient to reach a prefixed objective. For example, an agent could want data from another agent in different way respect the standard protocols. For this reason, an agent has to manage its characteristics (e.g., output, external interface, internal protocols). This property is particularly important in industrial systems because they are prone to dynamic changes in management and technical levels.

### 3.1.5. Social Ability

An IA has to interact with other agents (also humans). This feature is the core of the intelligent agent theory; through the interaction an agent can understand the events and adapt its characteristics to the dynamic situations surrounding it. An IA needs to communicate with the internal and external environments to ensure several main activities (e.g., find out the state of other agents, sub-tasks synchronize, tasks scheduling, acknowledgement). Furthermore, it has to be considered that some agents (e.g., mediator agents, contractor agents, negotiator agents) have particular assignments that, more than others, need to interact with the surrounding entities. In industrial systems this feature takes a key value in business plans and technical activities.

The last property introduces a main matter in the industrial systems, *i.e.*, the languages through which the agents exploit their own social ability. In general, the most important challenges in Agent Oriented Software Engineering (AOSE) is to use or create a suitable Agent Oriented Programming Language (AOPL) to allow the agents an effective and efficient interaction. There are several advanced tools that aid the developers in designing complex agent based software systems. Moreover, there are several powerful programming languages designed to facilitate the programming of agent entities. The chosen of the correct tool and language to achieve a prefixed assignment is a critical matter. Independently from the agent implementation there are some issues (e.g., debugging, validation, verification) that depend on the specific context in which the agents have to operate.

In AOPLs the focus is on how to describe the behavior of an agent in terms of constructs [11] (e.g., plans, communication strategies, interaction patterns, goals, messages). This agent description is aimed to specify the "reasoning state" of the agent during its activities. A typical example is AGENT-0 [12] a simple and powerful generic agent interpreter. This AOPL tends to program agents in terms of basic behavior, in this way the properties of the agents can be developed by definite notions, such as: Reactions, communication ways, interactions, rules, and so on. Another interesting AOPL is dMARS (Distributed Multi-Agent Reasoning System) [13], which provides both a sophisticated monitor and manageable macros to develop interaction activity between agents. Moreover, this AOPL provides several functionalities to manage critical implementation steps (e.g., configurations of internal states, perception of events). The dMARS can be considered a reference technology to direct complex industrial systems since it has a wide range of high solutions that support the whole industrial processes. A last interesting AOPL considered to develop agents in industrial systems is JACK [14]. It is a versatile framework designed to create, by models, the concepts that drive the intelligent agents. This language has a wide use in industrial contexts since it provides powerful tools for the run-time support. Moreover, JACK has predefined structured patterns to manage critical situations (e.g., concurrency, reaction to events, failure). In the last years, there have been many efforts to support the IA communication in different fields, including the Industrial systems, the result has been a growing development of multi-purpose languages (e.g., 2APL, 3APL) [15]. An agent has several others features that can be considered implicitly included in the previous features (e.g., learn capability, reasoning, auto-improvement).

In complex industrial systems there are several kinds of agents according to different processes or tasks. For

 

example, the communication-agents are specialized in communication-activities between sets of agents. They deal with strategies and protocols for the information or data exchange. Another example is the learning-agents that learn about the functionality and potentiality of the surrounding environment and transmit this knowledge to the other agents. Several others dedicated agents are developed according to specific requirements. Beyond the potentiality of a single IA the algorithms that drive the industrial processes are multi-agent based.

MASs can be considered as composed by heterogeneous agents, which are aimed to achieve common objectives. Likewise to the intelligent agent definition, also in this case there are several similar definitions of MAS [16]. In our context a MAS can be described as a suitable and reasoning assemble of agents that, according to their features, achieve common objectives through connections and teamwork intelligent mechanisms. Also the MASs have the same features observed for the intelligent agents: the main difference is the coordination and cooperation processes that drive the agents. Moreover, a MAS has interaction properties that define the collective reasoning of the whole system. Usually, each MAS has a set of rules and protocols that are inherited from each single agent. These last define the kind of collective intelligence characterizing each single MAS. These considerations conclude the paradigm definition of MASs.

A complex MAS can be composed by different sets of MASs obtaining a system with high levels of communication processes. In any case, these systems are characterized from more complex coordination, cooperation and teamwork mechanisms. Usually, no centralized control methods are implemented, but specific strict rules and hierarchic behavior strategies are performed to ensure the correct working of the system. The difficulty in the MASs management depends on the number of agents and their complexity. **Figure 5** shows a typical MAS

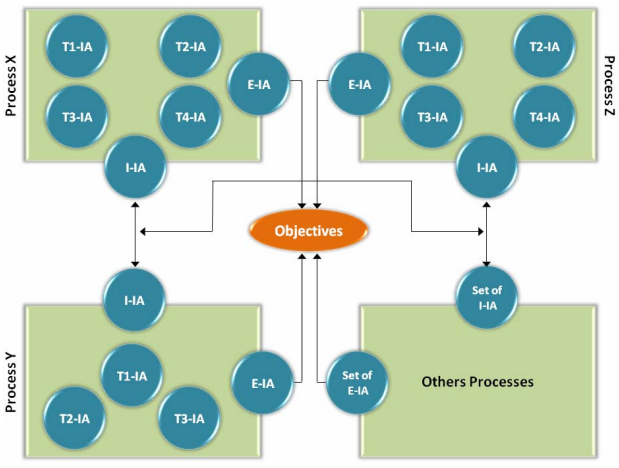


**Figure 5. Simple MAS scheme.**

scheme where each process is driven by a set of agents that have different dedicated assignments.

Each process (X, Y, Z and others) is composed by some tasks that are accomplished by several agents. As presented in **Figure 5**, the process X is composed by four industrial activities (T1, T2, T3 and T4, where T = Task), and two service activities (E and I, where E = Executor, I = Interface). All the processes involved in the industrial environment, independently from their role, are aimed to reach the common target: the objective (artifact/activity).

The agents tied to the industrial activities accomplish specific tasks tied to the industrial process, while the others two activities provide coordination and cooperation mechanisms. I-IA and E-IA work on the whole process. In fact, the first provides the result of the Industrial process (e.g., artifact or part of it, data, information) to the environment; the second uses a communication channel to interact with other processes (*i.e.*, with the related I-IAs). Note that only the I-IA and E-IA can make actions outside their own process. The other four agents (T1-IA, T2-IA, T3-IA and T4-IA) can accomplish only internal assignments and can interact only with agents belonging to the same process.

## 4. MASs: Approches and Algorithms

The new generation of the industrial systems can be conceived like Distributed Artificial Intelligence (DAI) systems described as cooperative systems where a set of heterogeneous agents act jointly to solve a given problem [17]. A DAI system belongs to the Distributed Problem Solving (DPS) field where a problem is solved by using several modules [18] which cooperate in dividing and sharing knowledge about the common issue. All these factors influence the choice of the multi-agent architecture to drive the algorithms used into each agent and the related communication strategies. There are several multi-agent architectures used to perform the activities of a large set of heterogeneous environments; the most effective are inspired by the distributed version of BDI (Belief-Desire-Intention) [19]. This architecture is based on the correspondence between beliefs, desires and intentions with reciprocal identifiable data structures. The BDI based systems have several features (e.g., agent scalability, real-time communication, acknowledge mechanisms). They can be managed to be adapted in any environment. In **Figure 6** a general overview of the basic patterns to implement industrial architectures is given. They are based on the classical configurations used in others fields, the differences are came out during the implementation by considering the features of each agent in accordance to the specific industrial system.

As a general rule, each task is assigned to a specific Task Master Agent (TMA) which, according to the spe-

 

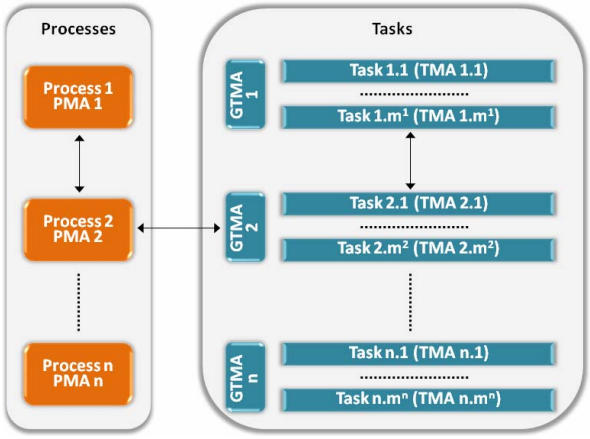


**Figure 6. Sample of architectural industrial system.**

cific architecture, can solve different assignments (e.g., safety and security, execution, coordination, cooperation). The TMA can work alone or supported by several sub-agents, depending on the nature and complexity of the task. These sub-agents deal with only a peculiar and homogeneous step of the task. Moreover, they are usually execution-oriented (*i.e.*, executor agents) or information-oriented (*i.e.*, acknowledger agents). Commonly, the sub-agents can have relationships only with sub-agents of the same task or with the related TMA. A particular kind of TMA, the Gateway Task Master Agent (GTMA), has to interact with all the TMAs belonging to a same process. Finally, the Process Master Agent (PMA) is used to allow interaction of different agents. Each GTMA can interact only with the related PMA, each PMA can interact with any other PMA. The communication activities are usually performed by well-known protocols and algorithms in the network field. Indeed, each communication, in the MASs, tends to be implemented according to FIPA specifications [20]. It is the IEEE Computer Society organization that promotes agent-based technology and the interoperability of its standards with other technologies.

The MASs communication activity can be analyzed by graph theory [21], where each node is an intelligent agent and the edges between nodes are the communication channels. This allows to adopt the well-known algorithms to manage environments composed by multi-agent systems. The current MASs communication processes are inspired to the basic algorithms on network architectures [22]: **Routing**, **broadcasting**, and **semi-group computation**.

The **routing algorithms** regard the processes used to visit a graph to reach a particular objective. In literature exists different algorithms to accomplish this task [23]. For example, the Dijkstra and Bellman-Ford algorithms research a shortest-path from a single source node to any other node in the graph. A variant of these algorithms is given by Floyd-Warshall and Johnson that provide strategies to research a shortest-path from multiple source nodes. Also the Prim and Kruskal algorithms (minimum spamming tree) and the Ford-Fulkerson and Karp algorithms (maximum flow) are commonly used in routing issues. All these algorithms are used to allow the agents different communication strategies which are tied to several factors, such as: type and aim of the interaction, kind of activity, kind of involved agents, and so on.

The **broadcasting algorithms** provide a direct way to communicate to a node in a graph. These algorithms are based on one-to-all philosophy, where a single node (*i.e.*, agent) has to send (*i.e.*, interact) a message to all the nodes of the graph. The weighs in the graph can drive the communication process according to the specific objective. They are usually represented by vectors that include a wide set of information (e.g., priority access value, identification value (ID), agent state value). In industrial contexts these algorithms are used to support control activities and to provide commands and scheduling steps to technical processes.

The **semi-group computation algorithms** are based on the binary communication of the intelligent agents composing a definite and closed set of agents. In general, the binary communication is used to reach each node in the graph through a dynamic bridge that allows two agents to communicate according to some convenience rule (e.g., distance between the agents, cost of the connection, priority level). These algorithms are usually applied, in industrial environments, where a strict intelligent agent structure is required. The particular structure of a semi-group of agent is suitable to implement systems able to react to non-deterministic events (e.g. external operations, unexpected broken events, sabotages). In addition to the three mentioned basic algorithms (and their evolutions) on network architectures, there are several experience-based approaches for everyday use which provide ad-hoc solutions allowing a qualitative improvement of the management standards. In this way, the human experience can be utilized to optimize the agent implementation improving the industrial process.

**Table 1** presents a brief description of the communication algorithms according to the different main industrial environments. The physical connection of a MAS is a crucial point of the current and next generation Industrial environments, in fact the way in which sets of agents are connected each other can drive the chooses about the adopted interaction communication technologies. For example, in large-scale management processes centralized and hierarchical architectures are preferred. On the contrary, in advanced manufacturing processes are used flexible, scalable and modular architectures. **Figure 7** shows that the architectures for industrial systems are

 

**Table 1. Communication algorithms.**

| Industrial Environments | Algorithms | | |
|---|---|---|---|
| | **Routing** | **Broadcasting** | **Semi-Group** |
| **Management and Strategies** | general spamming tree | one-to-all one-to-set one-to-one And vice versa | binary computation |
| **Supply Chain Management** | maximum flow shortest path | one-to-all set-to-set and vice versa | binary computation linear computation |
| **Control and Driving Tools** | minimum spamming tree maximum spamming tree | one-to-all one-to-set and vice versa | binary computation multiple computation |
| **Security Activities** | multiple shortest path general spamming tree | one-to-one one-to-all and vice versa | multiple computation |

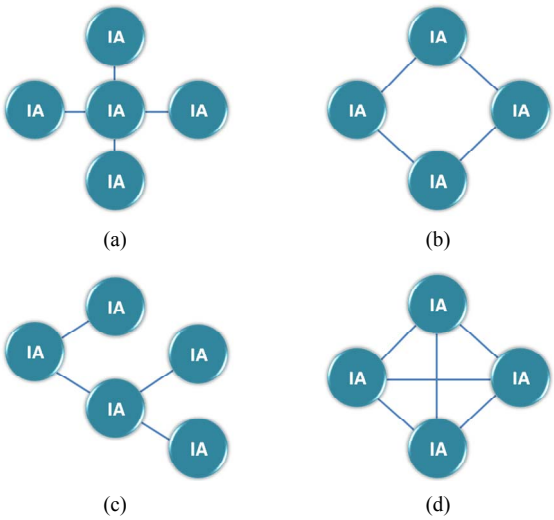


**Figure 7. (a) Star; (b) ring; (c) chain; (d) network structures.**

implemented with four approaches: Star, ring, chain and network.

The *star-structure* is used in processes, tasks and subtasks where a centralized mechanism is needed. The activities are managed by the central agent that can directly interact with each node in the graph. A typical example of this architecture regards the interaction between the interface intelligent agent (*i.e.*, I-IA, the central node) and the other agents belonging to the same task. A different approach is provided by the *ring-structure* where a decentralized interaction mechanism is given. The responsibilities and the activities are subdivided between the agents involved in the graph. A typical example is given in the processes managed from a large amount of heterogeneous agents. The *chain-structure* provides a hierarchical interaction structure. This approach is commonly used to allow the communication between different levels. This interaction is the basic communication approach used in the industrial systems where every level has to solve different assignments. Each node can interact only with its own father or sons. The *network-structure* provides an interaction mechanism where each node can directly interact with any other. In this architecture the nodes have similar assignments. It is usually used to allow the agent of a same level to interact each other.

In our context, an industrial process can be classified within one of the following classes: Planning and Strategies of Management Processes (PSMP), Driving and Control of Manufacturing Processes (DCMP), Advanced Analysis of Critical Processes (AACP). The PSMP term highlights the high level activities that drive the whole business processes. These activities cover each level of the industrial architecture and include a wide range of assignments (e.g., economical strategies and planning, coordination of internal and external activities, definition of policies and rules). A classical application of the MASs regards the Supply Chain Management (SCM) [24]. A supply chain is a network that deals with several activities (e.g., materials, services, logistic planning) to reach the distribution of a final artifact. The PSMP are usually directed by statistical and computational algorithms. Each agent in the system controls a specific algorithm. The aim of the agent is to elaborate the information about the environments to obtain numerical values and, then, computes the values through statistical models to obtain a numerical transposition of the real world involved in the coordination activities. An interesting kind of algorithms regards the deduction/cognition process. Usually, these algorithms are performed by a mathematical-agent that knows linear programming and polynomial approximation techniques, useful to derive deductions. Other classes of algorithms are available according to the specific process.

Other important algorithms included in this processes class regard those to solve conflicts. The agents that accomplish this kind of algorithms exploit the social ability feature. In particular, fuzzy logic approaches [25] are implemented into the agent to face different issues (e.g., reliability of events, flexibility, mediator activity). **Table 2** shows a brief description of the common algorithms used in PSMP class.

The DCMP class regards all the activities concerning the manufacturing processes. This class deals with assignments, such as: Robots collective control, industrial tools management, technical safety policies, and so on. A classical application of this class regards the manufacturing robots control [26,27]. These algorithms regard the real-time reactivity on runtime environments. In fact,

 

they must guarantee an immediate and solving interacttion with the surrounding context. Indeed, in this class it is possible to distinguish two sub-classes of algorithms. The first regards the algorithms inside the industrial tools (e.g., firmware or machine level), the second is about the algorithms inside the technological structures of the manufacturing environments (e.g., workstation, data processing centre). The algorithms belonging to the first sub-class are designed to be independent and light. These algorithms do not have a high computational level, but they have more parallel mechanisms to avoid and overcome bad events (e.g., failures, accidents). The most important agent is the interface-agent that controls and manages the interaction between the tool and the external world. In the second case the implemented algorithms are statistic-based. In particular, a large amount of control algorithms, and related agents, have knowledge about the discrete and continue random variables. In this way, for example, a prediction about a lot of targets can be done (e.g., life of a component, the state of an entity, the safety state of a tool). An interesting aspect related to the second sub-class of algorithms regards the simulation environment for manufacturing activities. These algorithms are used in modeling and simulation environments for improving or testing the different levels involved in the manufacturing activities. The algorithms used in this context regard the model understanding area (e.g., probabilistic models, prevision models). These algorithms are used to check the planning and scheduling strategies for a specific operative task. Moreover, they allow to the user to achieve simulations and predictions and to obtain different information about the events that occur during the plan strategy [28]. Also in this case **Table 3** shows a brief description of the main DCMP algorithms.

The last class of processes, AACP, is referred to critical tasks, sub-tasks and processes. This class considers each level of the industrial system, from technical up to economical issues. An interesting example is provided by the monitoring and diagnosis systems applied to sophisticate manufacturing environments (e.g., production of chips, production of complex artifacts, production of components, model testing). In this case the system provides several run-time dependent functionalities that perform activities as monitoring and diagnosis. These functionalities are accomplished by advanced algorithms (e.g., sensor understanding, pattern recognition, machine learning). If an agent has some doubts regarding the correct execution of a task, it may ask to interrupt each activity of the process. For example, particular agents can be created to recognize particular features on images that represent welding on an artifact in control process. The agents work on the images and they give information about the quality of the welding to the industrial manufacturing system. **Table 4** presents the main algorithms belonging to the AACP class.

**Table 2. PSMP algorithms.**

| PSMP | Algorithms |
|---|---|
| **Economical Planning** | deduction and reasoning, simulation, exhaustive research, linear approximations |
| **Supply Chain Management** | combinatory, linear programming, polynomial approximation, fuzzy logic |
| **Strategic Planning** | prediction, heuristic, comparison, numerical evaluation and synthesis |
| **Stock Management** | binary research, dictionary management, balanced trees, fuzzy logic |

## 5. Hybrid Systems: A Case Study

An interesting opportunity regards the use of more than one computational intelligent (CI) approach to solve issues in industrial systems. For example, in [29] is described a DNA-MAS genetic programming system, which is designed for application-generic multi-agent simulation and generation using an advanced symbolic language built specifically for the use in a random mutation and crossover environment. Indeed, in industrial systems, the other techniques of artificial intelligence (e.g., genetic algorithms, neural networks) are used to improve specific features of an agent. For example, in a manufacturing environment, where the visual images of the artifact are important for the quality control, an agent could need to exploit the potentiality of a genetic algorithm (which approximates a set of solution to the optimal solution) to try a better identification of the meaningful features of the same images. There are several examples, not only in industrial systems, where hybrid

**Table 3. DCMP algorithms.**

| DCMP | Algorithms |
|---|---|
| **Driving** | real-time tracking models, interactive patterns, experience-based models |
| **Control** | discrete and continue random variables, prediction models, reasoning-based models |
| **Runtime** | real-time acknowledgment, scheduling of dynamic queues |
| **Simulation** | probabilistic models, prevision models, non-deterministic models |

**Table 4. AACP algorithms.**

| AACP | Algorithms |
|---|---|
| **Control** | pattern recognition, deforming models, feature extraction, global and local operators |
| **Position** | tracking, scheduling, machine learning, feedback ranks, feedback management |
| **Driving** | feedback management, retroactivity strategies, experience-based models |
| **Simulation** | prediction, stochastic computation |

 

architectures can result suitable to face critical problems. The purpose of this brief section is to highlight that the artificial intelligence techniques have a common root (*i.e.*, the intelligent dynamic reasoning) that can be applied using different techniques to adopt a different representation of the human reasoning for solving activity.

An interesting case study, shown in [30], regards a hierarchically organized multi-agent system for production control of semiconductor wafer manufacturing facilities (wafer fabrication). The production control of wafer fabrication is challenging from a complexity and coordination point of view. The goal of semiconductor manufacturing is the production of integrated circuits. This environment is characterized by several industrial tools that are managed, in different way, from a large amount of users. Moreover, the semiconductor manufacturing domain is characterized by stochastic events like machine breakdowns and the change of customer related due dates of the lots. A flexible representation of the process conditions of semiconductor manufacturing domain has been defined with the purpose to develop the MAS architecture. Then, it has been provided modeling capabilities for agent hierarchies; moreover it has been provided to the system the capabilities to emulate a wafer fabrication represented by a discrete event simulation model for performance assessment of our agent-based production control system. Finally, it has been given a mechanism about integration capabilities for legacy software to use more advanced heuristics for staff agents. Through this implemented strategy an efficient complex industrial system has been developed.

## 6. Conclusion

The evolution of the current industrial systems and the challenges of the next generation ones are encouraging the development of new information management strategies to direct and manage the automated systems involved in the manufacturing processes. The technology tied to the CI approaches, like IAs and MASs, seem to provide a valid solution to this kind of issues. This paper presents an overview of how efficient, autonomous and intelligent entities can support a new point of view on each and every aspect regarding a complex industrial environment.

## REFERENCES

[1] R. Lee, "Computer and Information Science, Studies in Computational Intelligence," Springer-Verlag, Berlin and Heidelberg, 2012. doi:10.1007/978-3-642-30454-5

[2] L. Benyoucef and B. Grabot, "Artificial Intelligence Techniques for Networked Manufacturing Enterprises Management," Springer-Verlag, London, 2010. doi:10.1007/978-1-84996-119-6

[3] D. Laha and P. Mandal, "Handbook of Computational Intelligence in Manufacturing and Production Management," IGI Publishing, Hershey, 2007. doi:10.4018/978-1-59904-582-5

[4] K. Hermann, "Distributed Manufacturing: Paradigm, Concepts, Solutions and Examples," Springer-Verlag, London, 2010. doi:10.1007/978-1-84882-707-3

[5] W. Xiang and H. P. Lee, "Ant Colony Intelligence in Multi-Agent Dynamic Manufacturing Scheduling," *Journal of Engineering Applications of Artificial Intelligence*, Vol. 21, No. 1, 2008, pp. 73-85. doi:10.1016/j.engappai.2007.03.008

[6] D. S. Kim, C. S. Kim and K. W. Rim, "Modeling and Design of Intelligent Agent System," *International Journal of Control*, *Automation*, *and Systems*, Vol. 1, No. 2, 2003, pp. 257-261.

[7] V. Graudina and J. Grundspenkis, "Technologies and Multi-Agent System Architectures for Transportation and Logistics Support: An Overview," *Proceedings of the International Conference on Computer Systems and Technologies* (*CompSysTech*'05), Varna, 16-17 June 2005, pp. IIIA.6-1-IIIA.6-6.

[8] S. J. Russell and P. Norvig, "Artificial Intelligence: A Modern Approach," 3rd Edition, Prentice-Hall Inc., New Jersey, 2010.

[9] C. Z. Janikow, "Fuzzy Decision Trees: Issues and Methods," *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, *Part B*: *Cybernetics*, Vol. 28, No. 14, 1998, pp. 1-14. doi:10.1109/3477.658573

[10] P. Stone and M. Veloso, "Using Decision Tree Confidence Factors for Multi-Agent Control," *Proceeding of the Second International Conference on Autonomous Agents* (*AGENTS*'98), Minneapolis, 10-13 May 1998, pp. 86-91. doi:10.1145/280765.280780

[11] D. Mitrović, M. Ivanović and M. Vidaković, "Introducing ALAS: A Novel Agent-Oriented Programming Language," *Proceeding of the International Conference on Numerical Analysis and Applied Mathematics*, 2011, pp. 861-864. doi:10.1063/1.3636869

[12] Y. Shoham, "Agent-Oriented Programming," *Journal of Artificial Intelligence*, Vol. 60, No. 1, 1993, pp. 51-92. doi:10.1016/0004-3702(93)90034-9

[13] M. D'Inverno, M. Luck, M. Georgeff, D. Kinny and M. Wooldridge, "The dMARS Architecture: A Specifiction of the Distributed Multi-Agent Reasoning System," *Journal of Autonomous Agents and Multi-Agent Systems*, Vol. 9, No. 1-2, 2004, pp. 5-53. doi:10.1023/B:AGNT.0000019688.11109.19

[14] M. Winikoff, "JACK Intelligent Agents: An Industrial Strength Platform, Multi-Agent Programming: Languages, Platforms and Applications," *Proceeding of the International Conference in Multiagent Systems*, *Artificial Societies*, *and Simulated Organizations*, Vol. 15, No. 2, 2005, pp. 175-193. doi:10.1007/0-387-26350-0_7

[15] K. V. Hindriks, F. S. De Boer, W. Van Der Hoek and J.-J. C. Meyer, "Agent Programming in 3APL," *Proceeding of Autonomous Agents and Multi-Agent Systems*, Vol. 2, No.

 

4, 1999, pp. 357-401. doi:10.1023/A:1010084620690

[16] H. Sevay and C. Tsatsoulis, "Multiagent Reactive Plan Application Learning in Dynamic Environments," *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*: *Part* 2 (*AAMAS*'02), Bologna, 15-19 July 2002, pp. 839-840. doi:10.1145/544862.544937

[17] N. M. Avouris and L. Gasser, "Distributed Artificial Intelligence: Theory and Praxis," Kluwer Academic Publishers, Norwell, 1991.

[18] S. Kovalchuk, A. Larchenko and A. Boukhanovsky, "Knowledge-Based Resource Management for Distributed Problem Solving," *Proceeding of the International Conference on Knowledge Engineering and Management*, Vol. 123, No. 2012, 2012, pp. 121-128. doi:10.1007/978-3-642-25661-5_16

[19] N. Ronald, "Modelling Pedestrian Behaviour Using the BDI Architecture," *Proceeding of the IEEE/WIC/ACM International conference on Intelligent Agent Technology*, Compiègne, 19-22 September 2005, pp. 161-164. doi:10.1109/IAT.2005.104

[20] M.-P. Huget, "The Foundation for Intelligent Physical Agents," 2012. http://www.fipa.org

[21] R. J. Trudeau, "Introduction to Graph Theory," 2nd Edition, Dover Publications, Mineola, New York, 1993.

[22] B. Parhami, "Introduction to Parallel Processing: Algorithms and Architecture," Plenum Press, New York, 1999.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithms," McGraw-Hill, New York, 2002.

[24] T. Mentzer, "Supply Chain Management," Sage Publications Ltd., Thousand Oaks, 2000.

[25] M. Mohammadian, "Designing Unsupervised Hierarchical Fuzzy Logic Systems," *Machine Learning*: *Concepts*, *Methodologies*, *Tools and Applications*, IGI Global, Hershey, 2011, pp. 253-261.

[26] H. Colestock, "Industrial Robotics," McGraw-Hill, New York, 2005.

[27] D. Patrick and S. Fardo, "Industrial Process Control Systems," 2nd Edition, The Fairmont Press, Lilburn, 2011.

[28] S. Kraus, "Strategic Negotiation in Multiagent Environments, Intelligent Robotics and Autonomous Agents," The MIT Press, Cambridge, 2001.

[29] S. J. Kollmansberger and S. L. Mabry, "Intelligent Agent Generation with the DNA-MAS Genetic Programming System," *Proceeding of the International Conference on Artificial Intelligence and Soft Computing*, Banf, 17-19 July 2002, pp. 101-116.

[30] L. Monch, S. Marcel and J. Zimmermann, "FABMAS: An Agent-Based System for Production Control of Semiconductor Manufacturing Processes," *Proceeding of the* 1*st International Conference on Industrial Applications of Holonic and Multi-agent Systems* (*HoloMAS*'03), Prague, 1-3 September 2003, pp. 258-267. doi:10.1007/978-3-540-45185-3_24

 