Holonic Multi-agent Systems to Integrate Independent Multi-sensor Platforms in Complex Surveillance

J.J. Valencia-Jiménez & Antonio Fernández-Caballero
Departamento de Sistemas Informáticos & Instituto de Investigación en Informática de Albacete, Universidad de Castilla-La Mancha, 02071-Albacete, Spain
caballer@info-ab.uclm.es

Abstract

As far as a surveillance system is always integrated in an environment it has to adapt to possible changes that can occur in it. For this reason, it is not sufficient to install a series of sensors along the facilities to be guarded, as any modifications enormously increment the amount of data to be interpreted. Also, eventual failures or sabotages to the sensors produce the collapse of the system, which is not convenient at all in a potentially dangerous environment. Thus, the natural evolution of these systems is the integration in a compact system of intelligent platforms, which are able or not of moving in the environment, which possess several and complementary sensor types, and which interpret the information of each sensor coherently to offer the platform itself a fair service of surveillance. Quality of service is notably increased when there are a sufficient number of platforms forming a compact multi-agent system (MAS). Moreover, this MAS can itself be a compact subsystem of a superior hierarchy MAS composed of several subsystems. This is what we denominate recursive or holonic multi-agent systems.

1. Introduction

Thanks to the incorporation of distributed artificial intelligence [1], development of new technologies in detection (sensors and captors), robotics (actuators) and data communication enables surveillance systems to detect a wider frequency range, to cover a wider sensor area and to decide with a greater independence on the character of a particular situation.

No doubt that the main decision in any surveillance system is the positioning of the sensors in critical points that permit the optimization of the capacities of the sensors. A position can be static, when the sensor or platform is always in the same place [2], although it may consider different orientations; or dynamic [3], when a sensor is on a platform moving in the zone. When a platform carries different types of sensors (multi-sensorial platforms) [4] that complement each other to capture a portion of electromagnetic spectrum as great as possible, the overall information gotten is of a much better quality.

Anyway, communication among the system's elements is essential [5], as alarms have to be propagated along the subsystems and help or collaboration with other platforms is absolutely necessary to perfectly determine the situation and take the correct actions. If the multi-sensor platform is dynamic, the communication link should be wireless, whereas if the position is static, the link can either be wireless or wired. Nonetheless, wireless communication is preferred because the distance to other nodes may be very long [6].

Traditionally, control in an automatic surveillance system has been centralized, with a topology or configuration where sensors are like tentacles of a central node. The captured data is sent to the central node where it is processed and decisions on how to act are decided. This architecture is conceptually simple, but offers many problems related to scalability, bottlenecks and robustness, which are inherent to hierarchic rigidity of this architecture to not foreseen variations. For instance, this is the case when failures occur in the communication links, when sensors do not cover all areas in the environment, or when a burst of alarms collapses the control system and by domino effect the whole system.

Thus, it is necessary to consider new less centralized architectures, but not only in relation to the concept of distribution [7]. There are two main aspects in a distributed architecture that may help in complex surveillance systems. In first place, distributed systems allow the system nodes to possess a certain degree of autonomy to take their own decisions locally and to act independently of the central node. This way, problems derived from the temporal and spatial isolation of the nodes may be solved. Also, they can auto-adapt to changes or failures that are not foreseen, and there is less communication necessary in the system, helping to remove possible bottlenecks and enhancing the system's efficiency. Secondly, this kind of architecture enhances the performance of the system through the coordination of the distributed system's components. Consider, for example, the evaluation on the relevance of the events captured by the sensors, or the collaboration among the nodes for the resolution of a problem [8].

2. Holons vs. multi-agent systems

However, at the limit of a completely distributed architecture, a system with a predictable control, completely deterministic and with high performance - especially in a heterogenic and complex environment -, is not provided. A surveillance system needs reliable solutions in real-time and lacks in resources are not permitted. Thus, it is necessary to model the distributed system architecture in a suitable way. A very interesting solution is based in complex adaptable systems theory [9].

Arthur Koestler proposed the term “holon” [10] as a combination of the Greek word *holos* (all) with the suffix *on*, meaning particle or part of something. What moved Koestler to propose the concept of holon were two main observations: (1) First, it is easier to build a complex system when it is composed of elements or intermediate subsystems that are stable. Also, complex systems, like biological organisms or societies, are always structured as a hierarchy of stable subsystems at multiple levels, which again are recursively divided into subsystems of an inferior order. Structurally it is not a simple aggregation of elemental parts, and functionally it is not a global behavior like superposition of behaviors of elemental units; we are in front of a synergy. (2) Second, when analyzing hierarchies, systems and intermediate subsystems that are stable - in alive organisms and social organizations -, it is found that, although it is easy to identify “sub-all” and parts of the all, the “all” and the “parts” do not exist in an absolute sense, because they have a double nature of “sub-all/part”.

Holonic systems are systems that are modeled in terms of components (holons) that have their own unique identity but are part of a larger whole. This larger whole is known as a holarchy [11]. Holons are auto-content respect to their subordinate parts and simultaneously dependent parts when they are observed from higher hierarchic levels. Thus, the word holarchy denotes hierarchic organizations of holons; we are in front of a recursive structure. Holarchy can both guarantee performance stability, predictability, and global optimization of hierarchical control, and provide flexibility and adaptability of heterarchical control [12].

Therefore, according to this reasoning, a complex system can only be defined as such if and only if it is composed of other stable subsystems, where each one is able to adapt to events and to cooperate with other subsystems. This could directly be compared with a system of distributed hierarchy, with cooperative nodes. However, the key characteristic to introduce in the distributed hierarchy is the fact that subsystems cooperate in form of dynamic hierarchies. Temporary, to reach a global goal, they reorganize each certain time or when the global goal to be accomplished changes. This description of systems perfectly fits in multi-agent systems (MAS) theoretical and practical architectures. Indeed, a holon might be an agent or an autonomous multi-agent subsystem, if introducing the concept of “recursive agent”. And, the holarchy might be considered as the organization in a given instant of the hierarchy of agents, where all agents cooperate to reach a global goal and adapt in a dynamic way to the changes. Therefore, given the previous characteristics of the holarchy and the multi-agent architecture it is reasonable to consider the design of a complex surveillance system composed of multi-sensorial platforms as the design of a holonic multi-agent system, which reports considerable advantages at the implementation time, in terms of:

- **Bandwidth**: As holons inhabit the capacity of processing locally, they avoid the transmission of a great amount of redundant data to other holons, saving this way a great bandwidth in the communication. In our case, the saving in bandwidth would mainly belong to the data flow between sensors that catch the data and their later processing. There must be agents close to the sensors to receive and to handle this enormous amount of data and later be able to transmit the results through the interconnection network in a more efficient way.

- **Productivity or throughput**: The total processing capacity of the system is increased when holons participate in a parallel way, as it is the case in a system with distributed architectures. Here there should be different agent types based on the different parallel tasks to perform. For instance, we need agents for the management of sensor data and for the movements of the platforms.
- **Speed**: Distributed processing in holons not only increases the total throughput of the system as more charge is processed at the same time, but also

allows to process in less time the same load, since there is an equitable distribution of load among the system's holons. This way, a surveillance service close to real-time is achievable.

- **Robustness**: As the multi-agent system based on the holonic architecture is composed of stable subsystems in a recursive way, the tolerance to failures increases. Indeed, in addition to being able to have holons dedicated to the recovery and reconfiguration of the system, it is possible to start off from an initial stable situation at a given holonic level and to climb the level of the hierarchy in order to reach a stable state in the complete and global system. All agents execute valid solutions locally depending on the type of failure, and once an agent is recovered, it will coordinate with other agents; this way, the whole system is recovered.

- **Autonomy**: By definition holons are independent. In our system the multi-sensorial platforms must cover a potentially extensive area, and have to transmit the information of their surveillance to the other holons. It is physically necessary that multi-sensorial platforms are as independent of the rest of the system as possible, such that the unnecessary traffic that can occupy the transmission channels increasing the congestion probabilities of traffic and collapsing the system is kept to a minimum.

- **Scalability**: The holonic architecture is recursive; therefore subsystems can be integrated to build greater systems, which again are integrated in greater systems, and so on. A complex surveillance system may continuously integrate more and more multi-sensorial platforms. The system divides into subsystems that cluster platforms, the set of these clusters forms the system, and this way the system can grow up in an easier and trustworthy way because platforms are autonomous. Most of the processing load is local and the computational cost of synchronization and communication among agents can be assumed.

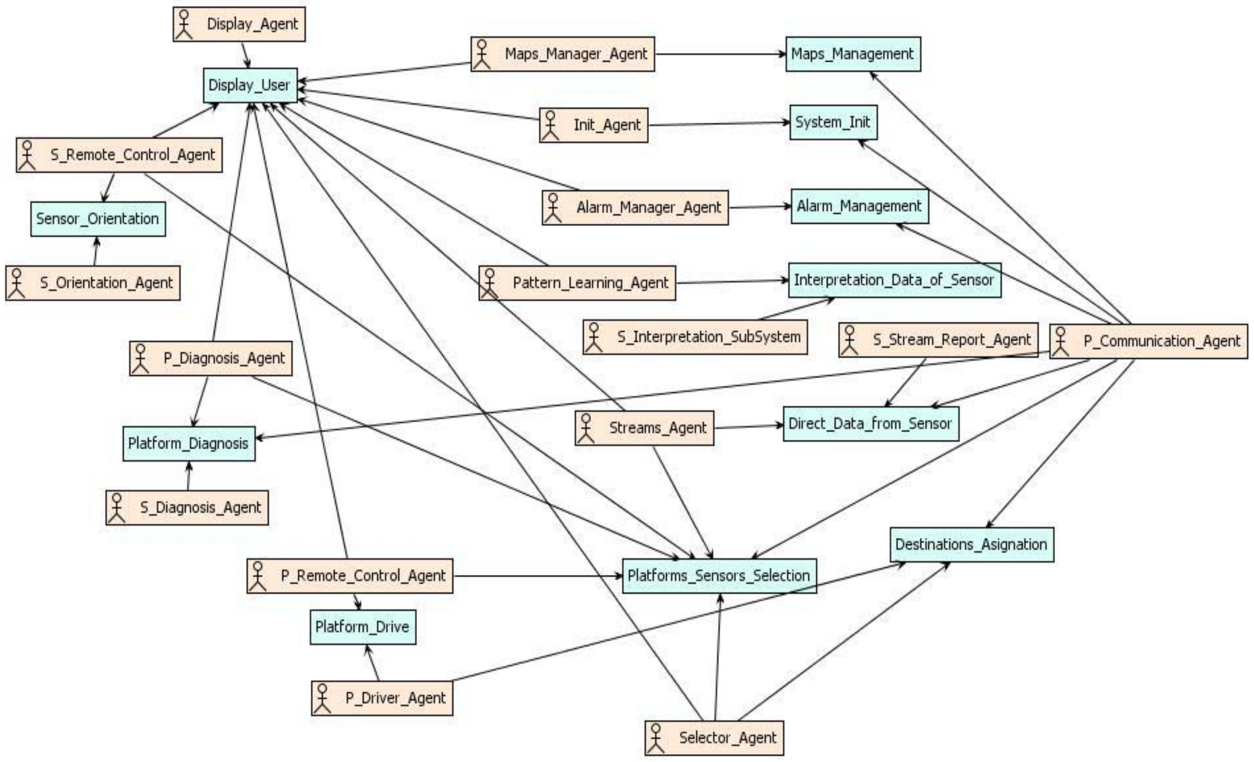

Figure 1. Agents and functionalities modeled in Prometheus

Next we introduce a preliminary design of a holonic MAS, based on the PROMETHEUS methodology [13], to make the surveillance system function with a series of multi-sensorial platforms.

3. An architecture for holonic multi-agent surveillance systems

The PROMETHEUS methodology basically consists of three consecutive and iterative phases [6]:

1. The system specification, which is concentrated in identifying the basic objectives, functionalities, scenarios and interfaces of the system, as well as in defining inputs (percepts) and outputs (actions).
2. The architectural design determines the types of agents of the system and their interactions.
3. The detailed design is focused in the contents of each agent and how the agents carry out the tasks.

In this paper we only include some aspects of the two first phases: objectives, functionalities and types of agents and their interactions. In addition, it is necessary to consider that the proposed MAS is indifferently made up of agents and multi-agent subsystems.

First, the principal goals that must be fulfilled by the basic system requirements are:

- The user is aware of wherever any platform is, what its operational state is, and any other information about the environment (e.g. obstacles, walls, stairs, holes, and so on).
- The user may select any platform and sensor in order to directly receive data, to move the platform or sensor, to assign new positions to the platforms (where they must move to), or to order a diagnosis of the whole platform and its sensors.
- The platforms are independent; they can move to other positions, watching along the way, and moving theirs sensors autonomously.
- The system must be able to provide an interpretation of the situation that is observed for each sensor, to decide if there is a threat or not.
- The system is capable of learning new parameters or patterns, so the sensors can make better decisions, lower the false alarm and non detection ratios.

In Prometheus functionalities are chunks of behavior that are formed by grouping related goals, jointly with related data, percepts and actions. Figure 1 shows a diagram including agents and functionalities. Thus, we obtain:

- *Maps_Management*: For creating and updating the maps stored in the maps database. The maps are used by platforms to move around the real environment.

- *System_Init*: To detect the communication network and, afterwards, all the possible platforms that are required to make a diagnosis, so the system updates the platforms database with information about the position and the state of all the platforms and their sensors.

- *Display_User*: The monitor displays all information about the map, position and state of all platforms and data from the selected sensors.

- *Platforms_Sensors_Selection*: The user selects platforms and sensors from the platform databases to apply further actions.

- *Platform_Diagnosis*: The platform checks its state and all its sensors every timeout or after the user launches the command; this information is communicated to the whole system. In order to increase the device-independent diagnosis, there is a diagnosis agent in the platform that controls the specific diagnosis agent for each sensor.

- *Destinations_Asignation*: The user selects platforms and assigns destinations to them directly or it is left to the system criteria (‘shortest way’); the platforms handle the routes autonomously.

- *Direct_Data_from_Sensor*: The user selects platforms and within these some sensors that send their data streams directly to be displayed on the user monitor. The user identifies the determined sensors that are in platforms, each sensor sends its data through the communication device of the platforms. Thus, there must be an agent in the platform that arbitrates the use of the communication if there are several sensors in the same platform sending data to the user’s monitor.

- *Platform_Drive*: Moves the platform independently or controlled by the user. At each timeout it communicates and updates the position of the platform on the map, avoiding the obstacles, walls and forbidden zones. Here, the configuration may vary enormously depending on the platform type and its propulsion type. It may be useful to assign an agent to each motor, axis or leg, and a superior agent to control them.

- *Sensor_Orientation*: Focuses the sensor independently or by means of the user control, communicating with the next functionality if something strange is observed.

- *Interpretation_Data_of_Sensor*: This functionality encloses all sensor processes of data capture and preproccessing, and latter recognition and interpretation of the recognized patterns (e.g. alarms). Due to the complexity of this task, this functionality completes a multi-agent system by itself.

- *Alarm_Management*: When a platform generates an alarm, the system reacts by sending a command to some platforms to confirm and to manage the potential threat by generating some plan of contingence.

After studying the functionalities for the surveillance system, holarchy is gotten by modeling three levels of agents. Firstly, we have defined the following types of agents for the overall system, including man-machine interface:

- *Display_Agent*: Manages maps, platforms, their states, and so on.
- *Init_Agent*: Contacts with the communication elements (stations, antennas), and afterwards with the platforms, asks for diagnosis, and creates the platform databases with their answers.
- *Maps_Manager_Agent*: Manages the maps database and which zones of the map are provided to each platform. When a platform exits its zone, the agent sends the zone that is necessary to the platform.
- *Selector_Agent*: Accesses the platform databases and selects platforms and sensors.
- *Streams_Agent*: This is an intermediate agent to *Display_Agent* to show the data of the sensors.
- *Alarm_Manager_Agent*: Coordinates when an alarm happens, and when necessary, it also elaborates contingency plans.
- *Pattern_Learning_Agent*: The user may specify how to update the sensor patterns databases with new details and parameters in order to interpret the situations.

Also, the platforms inhabit the following types of agents (second level in the hierarchy):

- *P_Driver_Agent*: Moves the platform.
- *P_Remote_Control_Agent*: When controlled by the user, it uses the *P_Driver_Agent* to move the platform.
- *P_Diagnosis_Agent*: Performs the diagnosis of the platform by asking for the diagnosis to all its sensors (*S_Diagnosis_Agent*).
- *P_Communication_Agent*: This agent is the most required one, as it controls all the communication of the platform with the system

Lastly, the sensors, at the last level of the holarchy, incorporate the following agents:

- *S_Orientation_Agent*: Controls the movement of the sensor.
- *S_Remote_Control_Agent*: The user controls the movement of the sensor through the *S_Orientation_Agent*.
- *S_Diagnosis_Agent*: Enables a diagnosis interface to update the types of sensors in the platform.
- *S_Interpretation_SubSystem*:
  - *S_Capture_Agent*
  - *S_Preprocessing_Agent*
  - *S_Pattern_Matching_Agent*
  - *S_Pattern_Learning_Agent*
- *S_Stream_Report_Agent*: In coordination with any other *S_Stream_Report_Agent* and the *P_Communitacion_Agent*, it receives the data of the *S_Capture_Agent* or *S_Preprocessing_Agent* to transform them into a stream sent to the monitor.

4. Conclusions

The interest to use agents in intelligent surveillance systems comes from their huge advantage when implementing very flexible and scalable complex systems. The paradigm of the holonic multi-agent systems introduced contributes to much more power at the time of designing systems that have to be integrated in most of the cases in a simple way into other subsystems that are currently working or that should work with a considerable degree of independence. This is precisely the case in the subsystem dedicated to the recognition and interpretation of the data captured by the sensors. Also, for the surveillance system it is much easier to integrate it and to belong to other greater systems, forming a constellation of stable systems.

As described in the paper, our approach benefits from an enhanced bandwidth, and better throughput, speed, robustness, autonomy and scalability. All this is of a great interest in a complex multi-sensorial surveillance system.

5. Acknowledgements

This work is supported in part by the Spanish CICYT TIN2004-07661-C02-02 grant and the Junta de Comunidades de Castilla-La Mancha PBI06-0099 grant.

6. References

[1] P. Remagnino, A.I. Shihab, and G.A. Jones, "Distributed Intelligence for Multi-camera Visual Surveillance", *Pattern Recognition*, Volume 37, Issue 4, 2004, pp. 675-689.

[2] X. Yuan, Z. Sun, Y. Varol, and G. Bebis, "A Distributed Visual Surveillance System", *IEEE Conference on Advanced Video and Signal Based Surveillance* (AVSS'03), 2003, p. 199.

[3] M.T. López, A. Fernández-Caballero, M.A. Fernández, J. Mira, and A.E. Delgado, "Visual Surveillance by Dynamic Visual Attention Method", *Pattern Recognition*, In press, 2006.

[4] R. Stiefelhagen, K. Bernardin, H.K. Ekenel, J. McDonough, K. Nickel, M. Voit, and M. Wölfel, "Audio-visual Perception of a Lecturer in a Smart Seminar Room", *Signal Processing*, In Press, 2006.

[5] A. Saad, and D. Smith, "An IEEE 1394-Firewall-Based Embedded Video System for Surveillance Applications", *IEEE Conference on Advanced Video and Signal Based Surveillance* (AVSS'03), 2003, p. 213.

[6] A. Mahapatra, K. Anand, and D.P. Agrawal, "QoS and Energy Aware Routing for Real-time Traffic in Wireless Sensor Networks", *Computer Communications*, Volume 29, Issue 4, 2006, pp. 437-445.

[7] S.-N. Lim, L.S. Davis, and A. Elgammal, "A Scalable Image-Based Multi-Camera Visual Surveillance System", *IEEE Conference on Advanced Video and Signal Based Surveillance* (AVSS'03), 2003, p. 205.

[8] J.M. Molina, J. García, F.J. Jiménez, and J.R. Casar, "Surveillance Multisensor Management with Fuzzy Evaluation of Sensor Task Priorities", *Engineering Applications of Artificial Intelligence*, Volume 15, Issue 6, 2002, pp. 511-527.

[9] N. Honma, K. Abe, M. Sato, and Hiroshi Takeda, "Adaptive Evolution of Holon Networks by an Autonomous Decentralized Method", *Applied Mathematics and Computation,* Volume 91, Issue 1, 1998, pp. 43-61.

[10] A. Koestler, *The Ghost in the Machine,* Arkana Books, 1971.

[11] J. Jarvis, R. Rönnquist, D. McFarlane, and L. Jain, "A Team-based Holonic approach to Robotic Assembly Cell Control", *Journal of Network and Computer Applications*, Volume 29, Issues 2-3, 2005, pp. 160-176.

[12] B. Huang, H. Gou, W. Liu, and M. Xie, "A Framework for Virtual Enterprise Control with the Holonic Manufacturing Paradigm", *Computers in Industry*, Volume 49, Issue 3, 2002, pp. 299-310.

[13] L. Padgham, and M. Winikoff, *Developing Intelligent Agent Systems: A Practical Guide,* Wiley, 2004.