Data normalization and integration in Robotic Systems using Web Services Technologies

Jose Vicente Berna-Martinez
Department of Computer Science
University of Alicante
San Vicente del Raspeig, Spain

Francisco Macia-Perez
Department of Computer Science
University of Alicante
San Vicente del Raspeig, Spain

***Abstract*— The robotics is one of the most active areas. We also need to join a large number of disciplines to create robots. With these premises, one problem is the management of information from multiple heterogeneous sources. Each component, hardware or software, produces data with different nature: temporal frequencies, processing needs, size, type, etc. Nowadays, technologies and software engineering paradigms such as service-oriented architectures are applied to solve this problem in other areas. This paper proposes the use of these technologies to implement a robotic control system based on services. This type of system will allow integration and collaborative work of different elements that make up a robotic system**

Keywords-SOA; robots architecture; web serices; management and integration.

I. INTRODUCTION

Robotics has become one of the most active emerging areas in which converge a large number of disciplines [1]. One of the biggest changes has been the expansion of the environments where they are used, from industrial environments to service robots for professional use and or domestic environments [2]. This implies that the variety of robots has grown, the number of devices integrated has increased and diversified, the scenarios are now unpredictable, dynamic and open, and therefore the complexity and heterogeneity of the underlying information has grown. To provide a solution to this problem are being implemented proposals related to service-oriented software applications and techniques of software distributed over the Internet [3]. But because the elements that make a robot operate at different levels of technology (electro-mechanical elements, algorithms and software functions, neural networks, etc.), first standardization is required for all items, so you can see all and each of them from the same functional level. This paper presents the standardization of robotic elements as a service through a conceptual architecture based on ICT and widespread in E-Business, which enables the management of information flowing through various channels and sources of a robot. In addition to allowing homogenization of the devices involved in any robotic system also allows for standardized treatment of information, solving problems of integration of heterogeneous information, helping to define the information flows in a dynamic manner and allowing to overcome the problems caused by different frequencies and different processing requirements of work for those elements of a robotic system.

For the development of the proposal in the next section we make a brief tour of the background of major related work. Then the normalization of the components of a robotic system is presented and architecture for the transformation of elements in services is proposed. Afterwards an instantiation of the proposed architecture using Web Services to implement control systems of autonomous mobile robots are constructed, and finally the main conclusions and future lines of work are shown.

II. BACKGROUND

A robotic system consists of a set of elements that operate together to achieve a goal. The nature of these elements may be different (electro-mechanical components such as sensors or motors, software elements such as route tracing algorithms, integrated circuit or systems on chip (SoC) to implement neural networks or pattern recognizers, and so on). Moreover, these elements may vary over time to adapt to new circumstances, environments or requirements [4]. However, from a functional point of view, each of these elements can be seen as an entity that receives information, performs an action and produces results (these results can be data or may be an action on the environment). This mode of operation is similar to what we observe in the distributed software components that make up distributed applications [5], and so we can use a similar conceptual base to define each of the pieces that form a robotic system rather than seeing the robot as a rigid set of devices than should communicate between them. A centralized implementation is robust and efficient, but these applications lack the properties necessary for their maintenance, modification, modernization, adaptation or flexibility to change in the medium to long term. These deficiencies greatly influence the management of information, because changing a source of information (for example an ultrasonic sensor for a laser sensor) usually involves reconsideration or rescheduling of part or even the entire system [6].

Several proposals have emerged to provide these features. These works provide a common framework for the development of heterogeneous robotic systems using frameworks or tools like BABEL [6], CLARAty [7], LAAS [8], DAMN [9], which generally provide those features found in software distributed applications like flexibility, modularity, code reuse, management of production cycles, low-cost development, adaptation to change, and so on. However, these works make different proposals that develop technologies or frameworks that require learning and produces that specialists in robotics are away from the world of software engineering, although the world of software engineering is which provides the desired characteristics. These characteristics are:

- Control applications must be modular to allow code reuse and rapid development.
- The control logic must be independent of hardware. The hardware provides the possibilities, but the software develops the skills.
- Support for communications should be provided by framework in which is developed the system and details should be hided for implementation of intelligence of the robot.
- Components must be able to communicate asynchronously transmitting values. If the components use references cannot be distributed independently.
- The components must be able to be linked dynamically, using modules that are necessary even in runtime.
- Reactive techniques exploit the characteristics of the real environment
- Deliberative techniques allow us to infer knowledge that is not implicit in the environment

III. Normalization of Robotics Elements

In our work we propose to rely on widely available technologies and paradigms in the development of distributed software applications, specifically service-oriented architectures. For this it is necessary that each of the elements of a robotic system is provided as a service, and each service needs a support in the form of services container. A service container provides the suitable software infrastructure to deploy high-level functions on the devices. In this manner is the service that determines what function is developed and not the element or device in which is carried out. An architectural model widely used in the development of E-Business applications will be the basis for defining our services container, the architectural model of n-levels [5]. In Fig. 1 we can see the architecture of n-Levels which reflects the elements that incorporate the service container. Fig. 1-a we can see all the software elements that make up the service container. At the user level is allowed access as services (consumers) as well as from other external systems using a view controller. At the access level, SOA and working drivers are responsible for controlling the security aspects of access. At the business level drivers and business orchestration give proper access to specific functions to be deployed on devices (calculate a path, detecting obstacles, store information from the environment, convert the movements of each system element in the current position, and so on.), and finally at the level of resources, appropriate adapters will provide access to resources such as storage, possibility of simulation on various platforms, and so on. These components are based on a common middleware services that provide common support in a generic way, as security services, service orchestration, service notification or discovery services.

Fig. 1-b shows a simplified view of architecture, where you can more easily observe that the container will comprise a series of application components that define the specific functionality provided by the container, the middleware services layer common to all components and below the layer formed by the OS and the hardware specific to each device. Through this transformation, a motor is not a physical device with which the system has to communicate in specific and concrete way, but it becomes a service that can be consulted, to which we can transmit orders and can make decisions as launch an alert to another element of control when circumstances require.

The concept of service container is easily applicable to those robotic elements of computer nature, such as pattern recognition algorithms running on a computer to identify objects in an image. However, the electro-mechanical devices of a robot (such as motors or sensors) have no basis for processing, in other words, usually have no computational or transmission elements that allow communication with other

a) Full view b) Simplified

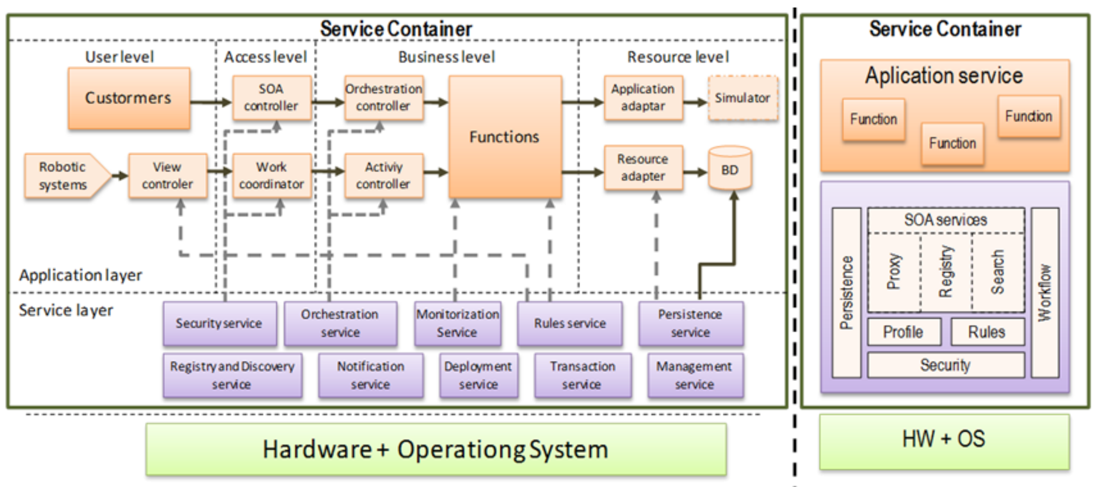

Figure 1. Full view of the elements that make up the service container architecture. b) Simplified view of the service

elements. To make these devices capable of computation and communication is necessary to convert the physical devices on smart devices. To do this it is possible to incorporate the hardware necessary to bring any physical device can become a service [10].

In recent years, advances in electronics and communications have given us a range of new devices that can provide such capabilities, so-called embedded devices. These devices are characterized by their small size and low cost, allowing its integration into other devices. Through these devices we can provide advanced functionalities to electro-mechanical devices that form a robot and introduce distributed computing paradigms.

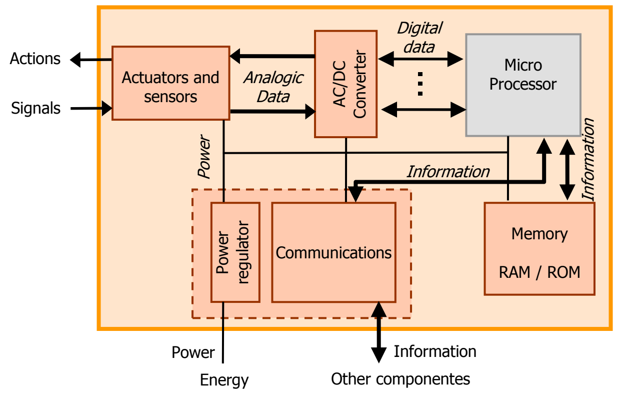

Figure 3. Physical structure of the control

Figure 2 describes the general structure of the embedded hardware. These items can transform a passive device like a motor on a device with computing capabilities. To do this, the unit requires embedded processing unit, AC / DC converter device to communicate with actuators or sensors, internal memory and a communications module that allows to interact with a network of devices.

In fig. 3 shows the embedded device selected for our proposal, the XPort device. XPort is a compact solution which includes a 16 bit processor, RAM, Ethernet port 10/100 and serial interface that allows communication with devices such as motors or sensors. This device has already been the subject of other studies in our laboratory [11] demonstrating that the physical characteristics are sufficient for the deployment of network services.

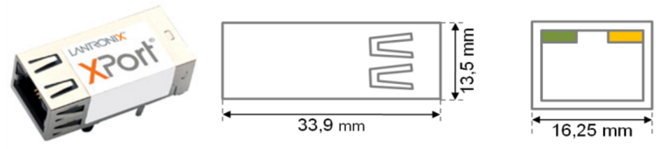

Figure 4. XPort device description used in our experiments

IV. Testing and Validating

For the instantiation of our architecture we rely on autonomous mobile robots. Mobile robots are particularly interesting when used in open environments because in these environments the quantity, quality and accuracy of information is uncertain. Other reasons to tackle this type of systems is that can be highly variable: legs, wheels, chains, several sensory systems or multiple algorithms for estimation of position, which means involving a greater or lesser number of computational processes.

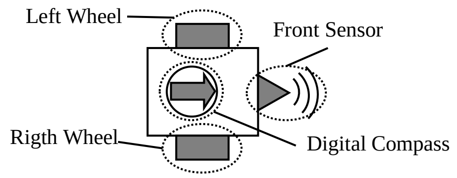

Figure 2. Physical scheme of robot: 2 wheels, a digital compass and a front sensor

In our work we have tried two behaviors: Behavior1 (B1) - navigating through the environment from a source point to a target point, and Behavior2 (B2) – navigating through the environment from a source point to a target point with obstacle avoidance. B2 will be implemented by adding new services in B1. For our system we used a generic robot equipped with two actuators (right wheel and left wheel) from which we get the current position of the wheel (shaft encoder sensor), a digital compass that indicates the current direction and a front-sensor obstacle detection (fig. 4).

In the functional analysis of behavior we have divided each of the functions of a robot in a service, isolating each function in an independent entity [12]. Each service is executed independently (fig. 5). B1 analysis produces the following services: Sensing, services responsible for monitoring the sensing devices; Interpretation, service responsible for translating the values obtained by the sensing to consistent data

service	description
Se	sensing service: right, left, compass
In	interpretation service: right, left, compass
Si	situation service
Re	reasoner service
Pl	planner service
Mo	motion service
Em	embodiment service
Ac	actuator service: right, left

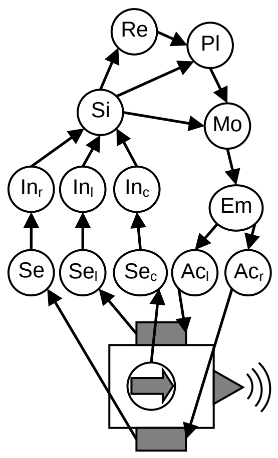

Figure 5. Decomposition of behavior 1 in services, each sensor has sensing service and interpretation service, each

(for example floating numbers to numbers with two decimal numbers); Situation, service responsible for using the data of Interpretation to obtain an estimate of the robot's position (in this case position in the environment, but it could estimate the position of the arm, relative position, etc.); Reasoner, service responsible for determining the mission to perform, in this case lead the robot from point A to point B; Planner, service responsible for planning the robot path; Motion, service which is responsible for obtaining the next move to be performed by the robot based on planning; Embodiment, service responsible for transforming the type of motion in terms of physical structure of the robot; Actuator, services responsible for managing communication with the actuating devices.

B2 analysis incorporates new services to B1. New services are shown in fig. 6: Sensing (Sed), control service for distance sensor, Interpretation (Ind) for the sensing service, a new service, Restriction (Rc), service responsible for calculating where the obstacles based on the interpreted data, and a new service Planner (Plo) which modifies the B1 planning for obstacle avoidance.

service	description
Se	sensing service: right, left, compass, distance
In	interpretation service: right, left, compass, distance
Si	situation service
Rc	restriction
Re	reasoner service
Pl	planner service
Mo	motion service
Em	embodiment service
Ac	actuator service: right, left

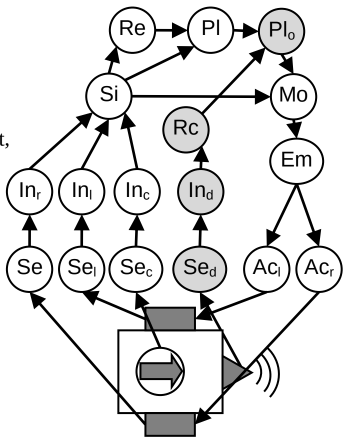

Figure 6. Decomposition of behavior 2 in services. The new services change the behavior B1.

Each of the services that integrate the control system develops a simple function, for example, Situation estimates the current position using techniques of odometry, Interpretation services translate the encoding axis of wheels into distances depending on the diameter of the wheels, and so on. Separate each system function in a service allows you to change services without changes influence the rest of the system.

For the implementation we used Microsoft Robotics Developer Studio (MRDS) because this environment provides us with an integrated development environment. NET for the design, execution and debugging robot applications scalable, concurrent and distributed, in addition to providing features such as service coordination, monitoring, configuration, deployment and reuse. RDS is built on two basic components: the Concurrency and Coordination Runtime (CCR) and the Decentralized Software Services (DSS). The CCR provides a programming model to handle multi-threaded applications and synchronization between tasks while the DSS allows to build applications based on a model of loose coupling. In addition DSS provides a lightweight model of state-oriented service that combines the concept of Representational State Transfer (REST) with a system-level approach for building high performance scalable applications [13].

In our experiments we used the simulator MRDS, a Lego robots and a homemade root, because it demonstrates the adaptability of the control systems based on web services to any type of robot, although its components are not the most accurate. Fig. 7-a show a view of the simulated robot composed of the elements described above, and fig. 7-b show a Lego robot equipped with the same real elements and fig. 7-c show the homemade robot with the same elements.

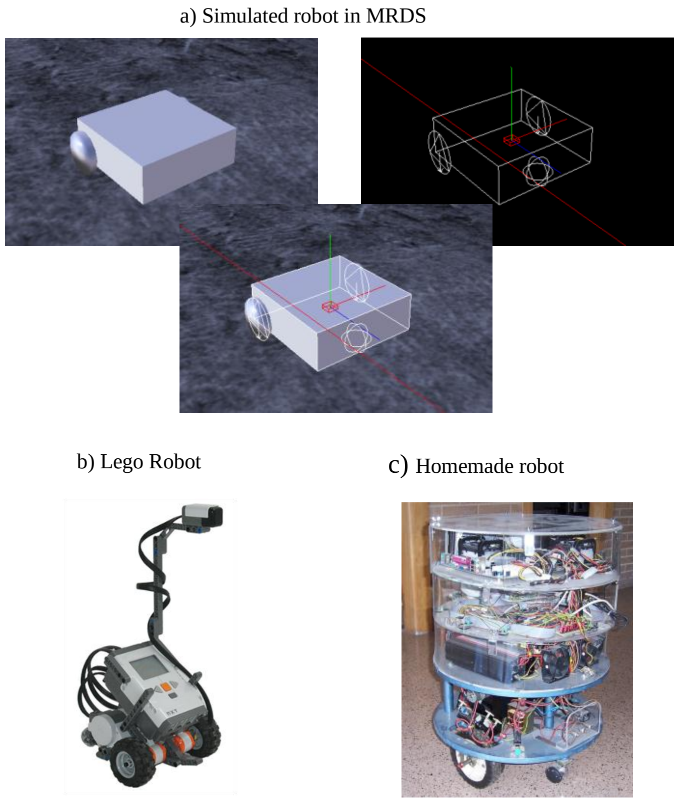

Figure 7. . a) Simulated robot. b) Lego Robot. c) Homemade robot

After deploying Web services and compose the control system according to the diagram in fig. 5 and fig. 6, we get the complete control system. For both B1 and B2, the system behaves as expected. Fig. 8-a shows the simulated robot's behavior and Fig. 8-b shows the Lego robot's behavior. Both systems use the behavior B2.

When we indicate a destination, the robotic system starts and progresses to reach the end point. Using B1, if there are obstacles in the path, the robot collides with them. Using B2, the system detects obstacles and modifies the path to avoid them. Both the simulated system as the real robot, behaviors are those specified. Most services remain common to all systems. Pass from a simulated robot to a real robot only involves modifying the services of Sensing and Actuator to connect to the appropriate resource. To use the behavior B2 only have to add the services specified in Fig. 2-b. The system thus shows its adaptability to change, flexibility to modify capabilities and robotic devices, the ability to reuse code, and so on.

The system has the peculiarity that each Web service operates at the frequency that requires its own characteristics. For example, the services responsible for monitoring each wheel require 50ms per cycle to obtain the state of the encoder. This data is transferred to the superior services but if this information does not imply changes (for example, the robot has not moved), Interpretation services will not produce new results. Similarly, the reasoning service starts the system when the current and desired position is not equal (not reached the destination) but during the execution will not release more

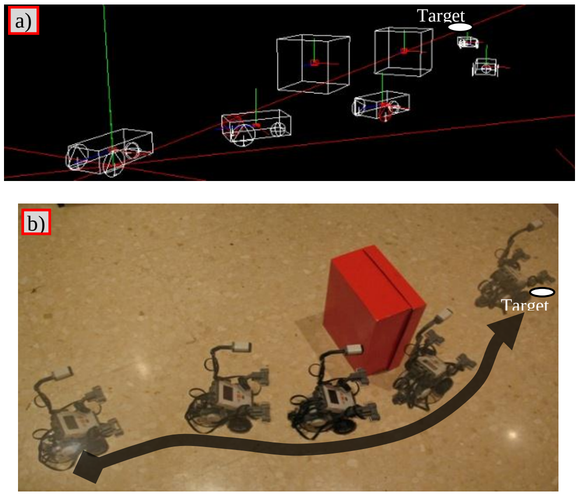

Figure 8. Simulated robot using B2 behavior. b) Lego robot using B2 behavior.

orders to planning services until it reaches the destination. Each service is independent, uses its own working frequency and its execution can influence whether or not the execution of other services and the communication is done homogeneously through message passing.

V. Analysis of Results

The experiments allow us to observe a number of features in the control system arising from the use of Web services:

- Each functional element of the robotic system has the same internal structure, all are equal.
- Each physical element of the robot is treated by the system in the same way, everyone is equal.
- The system is very flexible, can add and delete services even at runtime.
- The system is highly scalable, we can place each item in a different network node to run.
- We can reuse services or even share their implementation. For example, the obstacle detection services (Rc) may be used by other systems that require such information.

Each service can isolate units of information and its complexity, while enabling adapt each and every one of the types of information to a common message exchange. That is, and this is one of the most important feature, different and very different information / data is shared by the system, for example, information coming from different sensing devices, with different data types and different frequency. The system allows you to isolate each unit of data, adapt it and treat it without causing other negative effects on the system. All elements of the robot, now, run a common language of communication between them.

VI. Conclusions and Future Work

This paper has proposed the development of robotic control systems based on Web Services. This proposal allows us to standardize the elements of a robotic system and enables the exchange and processing of the information produced by each of the elements. It has also shown the implementation of this system for behaviors such as autonomous navigation without/with obstacle avoidance. The resulting system performs with the requirements and desirable features such as flexibility, adaptability, short development cycles, dynamics and absorption of problems of operating frequencies and integration and management of diverse information, regardless of the source and nature of the devices. Self-adaptation of the communication provides the perfect link between the computer functions and the physical system it controls.

We are currently working on two lines. In the short term we are increasing the range of services available: services to optimize the path of roads, services for environment mapping, management services for more sensors and actuators, and so on. In the medium term we are investigating the adoption of cloud computing technologies to move services to the cloud, so that the system be independent of physical resources.

References

[1] J.V. Berna-Martinez, F. Maciá-Perez, H. Ramos-Morillo and V. Gilart-Iglesias, *Distributed Robotic Architecture based on Smart Services*. IEEE INDIN 2006, ISBN 0-7803-9700-2, p. 480-485.
[2] IFR, Statistical Department, *World Robotics 2010 Service Robots*. 2010.
[3] S.L. Remy and M.B. Blake, *Distributed Service-Oriented Robotics*. IEEE Internet Computing. ISSN 1089-7801. Vol. 15 (2011), p. 70-74.
[4] J. Gowdy, *A Qualitative Comparison of Interprocess Communications Toolkits for Robotics.* Tech. report CMU-RI-TR-00-16, Robotics Institute, Carnegie Mellon University, June 2000.
[5] P. Harmon, M. Rosen and M. Guttman, *Developing E-business Systems and Architectures: A Manager's Guide*. Morgan Kaufmann Publishers, San Francisco, USA. ISBN 978-1558606654. 2001.
[6] J.A. Fernández-Madrigal, C. Galindo, J. Ganzález, E. Cruz-Martín and A. Cruz-Martín, *A software engineering approach for the development of heterogeneous robotic applications*. J. Robotics and Computer-Integrated Manufacturing. Vol. 24, issue 1, 2008. ISSN 0736-5845.
[7] R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras and H. Das, *The CLARAty Architecture for Robotic Autonomy*. Aerospace Conference, 2001, IEEE Proceedings. Volume: 1, pp. 121-132. ISBN 0-7803-6599-2.
[8] R. Alami, R. Chatila, S. Fleury, M, Ghallab and F. Ingrand, *An architecture for autonomy*. The Int. J. of Robotics Research, Vol. 17, No. 4, pp. 315-337 (1998).
[9] J. Rosenblatt: DAMN, *A Distributed Architecture for Mobile Navigation*. Thesis Doctoral, Tech. Report CMU-RI-TR-97-01, Robotics Institute, Carnegie Mellon University, 1997.
[10] J.V. Berná-Martinez, F. Maciá-Pérez, V. Gilart-Iglesias and D. Marcos-Jorquera, *Robotic architecture based on electronic business models. From physics components to smart services*. ICNCO 2006. ISBN 972-8865-60-0, p. 544-547.

[11] J.A. Gil Martínez-Abarca, F. Maciá Pérez, D. Marcos Jorquera, V. Gilart Iglesias. *Wake on LAN over Internet as Web Services*. Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'06). ISBN 1-4244-0681-1, p 1261-1268.

[12] J.V. Berná Martínez and F. Maciá Pérez, *Model of Integration and Management for Robotic Functional Components Inspired by the Human Neuroregulatory System*. IEEE ETFA 2010. ISBN 978-1-4244-6849-2.

[13] J. Kyle and T. Taylor, *Professional Microsoft Robotics Developer Studio*. Wiley Publishing, Inc. ISBN 978-0-470-14107-6.G. O. Young, “Synthetic structure of industrial plastics (Book style with paper title and editor),” in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

AUTHORS PROFILE

J. V. Berna-Martinez was born in Spain in 1978. He received his engineering degree and the Ph.D. degree in Computer Science from the University of Alicante in 2004 and 2011 respectively. Since 2006, he is an Associate Professor at the University of Alicante. His research interests are in the area of computer networks, distributed systems, bio-inspired systems and robotics which are applied to industrial problems.

Francisco Maciá-Pérez was born in Spain in 1968. He received his engineering degree and the Ph.D. degree in Computer Science from the University of Alicante in 1994 and 2001 respectively. He worked as System’s Administrator at the University of Alicante form 1996 to 2001. He was an Associate Professor from 1997 to 2001. Since 2001, he is an Assistant Professor and currently he is the Director of the Department of Computer Science and Technology at the University of Alicante. His research interests are in the area of network management, computer networks, smart sensor networks and distributed systems, which are applied to industrial problems.