Bigraph Theory for Distributed and Autonomous Cyber-Physical System Design

Vincenzo Di Lecce, Alberto Amato, Alessandro Quarto *Member IAENG*, Marco Minoia

***Abstract*—This paper proposes a multi agent system (MAS) implementing an innovative monitoring and control technique for industrial wastewater. Nowadays Cyber-Physical System (Cps) and Internet of Things (IoT) are becoming ever more common in this field. This fact led to the implementation of systems composed of many computing units intercommunicating mutually and characterized by computational power being adequate to the task to be carried out. The proposed MAS uses a cooperative approach among the various agents to achieve the global goal. The system knowledge is shared among agents. After an initial learning stage, the agents can cooperate to hit the global objective and eventually to self-assess the failure of some components. The system has been designed by means of the bigraph theory approach.**

***Index Terms*— bigraph, Cyber-Physical Systems, Internet of Things, model checking, monitoring, multi-agent system**

I. Introduction

CYBER Physical Systems (CPS) are considered an enabling key technology of Industry 4.0 because they can improve the growing of the three main pillars for the digitalization of the manufacturing sector (smart products, smart manufacturing and business models). An interesting aspect of CPS is the concept of "digital twin" that associates each physical device to its representation into the virtual world. So, each physical device is integrated with other electronic devices having computing, storage and networking capacities. This is the link between CPS and other powerful technologies, such as Internet of Things (IoT) and Multi Agent Systems (MAS).

IoT is a paradigm that allows for the interconnection and interoperability of everyday life objects, equipped with computing units, sensors, transceivers for digital communication and appropriate protocol stacks [1]. Nowadays, these digital communication systems can use many mature technologies such as Bluetooth, Near Field Communication (NFC), Radio Frequency Identification (RFID) for neighborhood devices and wireless network and 4G-LTE for far devices. On the other hand, these systems can produce a big quantity of data to be handled. This process is possible only with a digital infrastructure equipped by computational power not owned by the small processors used in these devices. Recently, a new computing paradigm has been emerging promising reliable services delivered through next generation data centers that are based on virtualized storage technologies: cloud computing [2]. This platform plays a key role in IoT approach because it can work:

- as receiver of data from the distributed sensors;
- as a computer to analyze and interpret the data;
- as a web based virtual interface providing services to various users.

Intelligent Agent technology is an important and a relatively new paradigm in software design. The term intelligent agent is now used as an umbrella term representing a wide range of software with different characteristics and abilities [3]. This fact led to many definitions of intelligent agent, but the authors agree with the definition proposed by Wooldridge and Jennings [4] stating that an intelligent agent is a problem-solving entity characterized by the following properties: autonomy, social ability, proactiveness and responsiveness.

The natural evolution of the intelligent agent technology is the Multi Agent System (MAS) technology [5]. Systems of this kind are composed of a set of intelligent agents interacting and collaborating with each other to solve complex problems that are beyond the individual capability or knowledge of each agent. In a MAS, agents can interact among them implementing a cooperative [6]–[8] or a competitive strategy [9], [10].

Using these technologies (IoT and/or MAS), it is possible to implement CPS working as autonomous networks of miniaturized intelligent sensors and actuators integrated into technical structures [11], [12]. Nevertheless, for practical and economic reasons, the hardware used into these applications is not too powerful (often small and relatively economic devices with low computational power, storage, power and network capabilities are used). This imposes strong constraints to the system designers because they must build ever more complex systems characterized by devices with poor resources with the target of intercommunicating so as to work as a unique and possibly autonomous system. The analysis and validation of these systems requires specific techniques just like bigraph as proposed in [12].

In this paper, authors describe a MAS implementing a cooperative strategy to achieve a global objective: filling a reactor with a mixture of liquid according to given percentages of the various components in a wastewater treatment plant. The proposed MAS has specific features that are presented in [4] and in particular it has a proactive behavior changing its strategies to react at the different external conditions. This system has been designed and developed as part of an innovative monitoring technique for

Manuscript received May 17, 2019; revised December 12, 2019.

V. Di Lecce is with DEI, Politecnico di Bari, Bari, 70126, Italy. (corresponding author, phone: +39-3701325542; e-mail: v.dilecce@aeflab.net).

A. Amato is with the DEI, Politecnico di Bari, Bari, 70126, Italy (e-mail: a.amato@aeflab.net).

A. Quarto is with the myHermes Srl, Taranto, 74121, Italy (e-mail: alessandro.quarto@myhermessrl.com).

M. Minoia is with DEI, Politecnico di Bari, Bari, 70126, Italy (e-mail: m.minoia@aeflab.net).

the control of industrial wastewater (website: http://maui.aeflab.net). This technique has been developed in the MAUI project under the grant of Lombardy Region (Italy). Bigraph theory has been used to design, test and validate the proposed system. The paper is organized as follows: section II gives a brief overview of the related work, section III describes the problem under analysis and section IV presents the proposed approach starting from discussing bigraphs and bigraphic reactive systems, while section V shows the case study analyzed in this research then section VI describes the experiments carried out and the obtained results. In section VII the conclusions and a perspective on future work are discussed.

II. Related Works

In the last years the technological development has resulted in the spread of miniature devices able to sense, compute and communicate wirelessly. Using this device, it is possible to build CPS acting as autonomous problem-solving systems. The concept of "digital twin" is the link between CPS and two powerful technologies: IoT and MAS. MAS has been used in various practical applications at different levels. For example, MAS technology was used in [13] to increase the reliability of a distributed sensor network, in [9] this technology was used for databases integration, in [6] as decision support system for an intelligent transport system, in [8] a MAS that implements mine detection, obstacle avoidance and route planning was proposed.

Due to the development of this kind of systems, architectural design techniques and exploration of complex software systems on embedded processor platforms are becoming increasingly popular. They are typically based on the study of communication or high-level modelling. This class of methods is based on models highlighting lack of flexibility at the system level both with pre-designed intellectual property cores and with most of the techniques for creating custom components. Another element of complexity is the reduced performance of these embedded systems.

Many of these architectural design tools are based on descriptive languages of the system's internal operations and exchanges. They involve multiple translation levels of the real problem into its representation that can be used to evaluate the achievable performance. Jansen and Bosch [14] proposed to see each software architecture as a set of explicit design decisions. In their idea (Archium), software architecture can be seen as a decision-making process and its design is about making the right decisions at the right time.

Márquez and Astudillo proposed [15] a model called COMPACT for the selection of components using some architectural tactics. Starting from the non-functional requirements, the proposed model follows these tactics to search for and identify the components suitable for the design. This model was successfully tested. Their tests highlighted the need to build a bridge between software architecture and system requirements.

Capilla et al. [5] proposed a web-based tool, which is capable of recording and managing architectural design decisions. Jansen et al. [16] developed a tool for making architectural decisions. This system supports architectural decisions semi-automatically and shares the results with all stakeholders.

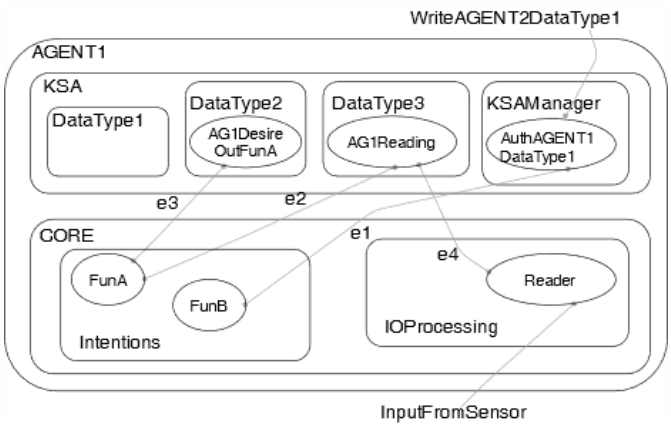

Fig. 1. An example of generic Agent represented through bigraph. This first version presents KSA while in the implemented MAS KSA is taken out.

In general, these approaches focus on documenting and archiving architectural states using certain tools. A similar situation is found in the design of IoT. Many authors use different kinds of software to represent architecture models based on various communication protocols, hardware components and central processing devices used in the system. The design compromise is between system performance and communication protocol performance (in terms of uniform and interoperable communication of the protocol itself).

Yu at al. [17] proposed to use bigraph technique as a tool to test context-aware applications. They built a data model based on the bigraphical meta-model and proposed to use the bigraphical sorted BRS to model context-aware environments. They generated the test cases to verify the interactions between context-aware environments and middleware along with domain services by tracing the interactions between the BRS model and the middleware model. The application environment typically includes a variety of physical structures, moving entities, and wired or wireless sensors connected to backend systems. They show the advantages of the model proposed and supported by a simulator. Furthermore, they highlighted that most of the simulation tools are mainly used to model and simulate the behavior of systems (such as Simulink [18] and SPIN [19]) and not to model the environments of applications.

Recently, UML class diagrams have been employed to model the static structures of the embedded real-time system (RTES) and its environment, Object Constraint Language has been used to model and automatically generate test data for RTES testing, and a state machine to model the behavior of entities [20].

Bigraphs and Bigraphical Reactive Systems (BRS) are graphical methods to describe the syntax and semantics of systems in terms of two orthogonal sets: connectivity and localization [21]–[23].

Some authors used bigraphs to control the models [10]. C. Tsigkanos and all. [24] proposed the use of these representations in the field of network access security (recognition of violations of the security requirements of a distributed access system).

III. Problem Description

There are many methods about the treatment of the wastewater, including the physical method, chemical

method, and biological method. For instance, the adsorption method with activated carbon is one of the most important physical methods. The organic pollutants in the wastewater are absorbed by the activated carbon. However, the adsorption method has the problem of saturation and secondary pollution. Biological treatment method with microorganism is also used to dispose the organic wastewater, but the efficiency of the biological treatment method is too low to be used for the disposal of a large amount of wastewater. As to regard to the chemical method, the settings of the ratio in the chemical compositions are of difficult level, this fact could imply an impact on the treatment effect of the wastewater. The complexity of physical, chemical, and biological phenomena associated with treatment units means that the performance of the process depends heavily on environmental and operational conditions. Moreover, the reactors show common features of industrial systems, such as nonlinear dynamics and coupling effects among the variables. These features explain the obstacles of the control strategies in wastewater treatment process (WWTP) i.e.:

- Dynamics [25]: Due to the inflow rate, water quality and contamination loads are undulated acutely; WWTP always runs in unsteady state. Moreover, these factors—inflow rate, water quality, contamination loads, pH, temperature, etc.—are passively received.
- Modeling Challenges [26]: The WWTP model consists of two main parts: the hydraulic model and the biochemical model. The hydraulic model represents reactor behavior, flow rates, and recirculation. The biochemical model is the second primary component of a WWTP model. However, it is still an open problem to seek accurate models to satisfy the complex characters of WWTP.
- Uncertainties [27]: Most bioreactors are equipped with sensors for online measurement of pH, temperature and so on. The appropriate strategy for achieving the control objective depends on the availability of online measurements such as biochemical oxygen demand (BOD) and chemical oxygen demand (COD). Unfortunately, most of the measurement techniques for these variables are limited to offline analysis in a research laboratory environment.

In this work, the authors propose a solution to handle the hydraulic aspects of the reactor behavior. In particular, a cooperative MAS has been designed to handle the process of filling a reactor with a mixture of liquid according to given percentages of the various components. This system has a proactive behavior changing its strategies to react at the different external conditions.

IV. From Bigraph and BRS theory to MAS modelling

Bigraphs and BRS can be used as a design tool for easy representation of MAS modelled cyber physical system (CPS) and interactions among agents and between agents and other system entities.

In this study, the MAS is composed of cooperative agents with local and global objectives that do not conflict with each others, as described in [28]. The MAS makes use of a private communication network able to guarantee to all agents the visibility of the environmental data (quantities read by all the sensors present, status of all the actuators present) that characterize the Knowledge Sharing Area (KSA).

Sensors and actuators can be physical or virtual. If physical, they will have the connection with an agent that is able to read data from sensors and/or controlling actuators. If sensors and actuators are virtual, they correspond to a value transmitted/received by other agents. In the proposed case study, the data are arranged in a numerical matrix in which each row corresponds to an agent and each column contains data of homogeneous type (e.g. column 5 contains the ambient temperature). This representation is the result of a formal project phase conducted by Bigraph's theory and BRS, as shown in the following section.

The first step in developing a MAS using BRS is to distinguish between subjects and objects in the domain model, i.e. which entities can perform actions, and which ones cannot [29]. This implies splitting controls into two sub-sets: controls for subjects and objects. As for the behavior of an agent and his internal status, the Belief-Desires-Intensions (BDI) approach is proposed in [10], [30]. This approach can capture both static and dynamic aspects of an agent and MAS.

The adoption of large graphical reactive systems for modelling the deployment architecture is described in [31]. This approach is suitable to meet various constraints of software components and target environments. The method uses a multi-scale modelling approach which provides for three sub-models: the execution environment, the software architecture and the integration model. The first two are bigraph and they are built by reaction rules.

In this work, the BDI paradigm has been used to model a CPS based on MAS as a bigraph. Furthermore, a new multi-scale approach is adopted instead of [31] to obtain the Bigraph that formally describes the system. This new method focuses on the construction phase of the bigraph on the agents and on CPS. BigMC tool has been used to support the bigraph building. This is an automatic testing tool for bigraphical reactive systems, used to build Bigraph and BRS and to verify the system properties [10].

A. The proposed approach

In this section the proposed approach to design a distributed CPS based on cooperative MAS using Bigraph and BRS is presented.

The common representation of MAS as BRS is not appropriate in the case of strong interaction between cybernetic space and physical space. To fill this gap, the state of each agent must be visible in real time to all other agents so that they have a total and unique knowledge of the environment. This representation is the base of the proposed KSA. BDI paradigm has been used to handle the knowledge sharing among agents. The state of each agent is constituted by beliefs and desires, which represent respectively what the agent knows about the environment and the desired state in which the agent or environment should evolve (also called goal or target [30]). The agent's status may also contain objectives to be achieved received from other agents. The agent who assigns a target to another agent must have the appropriate authorizations or must be hierarchically superior.

Each agent projects its state into KSA by writing the row of his competence, at the same time it works using data represented in KSA. To obtain an agent model suitable for

the distribution of a CPS, the authors provide an example of an initial prototype (Figure 1). This figure shows the structure of the agent that is formed by:

- CORE: this component manages the interaction between agent and environment/KSA. It receives stimuli from KSA and implements its intentions to produce decisions by performing actions to realize desires. Inside CORE there are two nodes:
  - the INTENTIONS node (I) includes all the procedures allowing the agent to make decisions and execute them. These procedures can be static or obtained from a learning process that observes KSA belonging to other agents. This node uses a subset of KSA information and updates another subset of KSA information.
  - The IOProcessing node includes the nodes for the management of the I/O of sensors and actuators.
- KSA: This component handles virtual information shared among agents. It includes agent status and manager for KSA operations, because the last one requires permission handling. Inside KSA there are two types of nodes:
  - DataType[n] nodes representing the agents' current status through information about their beliefs or desires.
  - The KSAMANAGER node specifies whether an agent can update a state belonging to another agent (e.g. to assign a desire) or not. It should be noted that the information from a sensor is processed and placed in the KSA by AGENT1 in the data class called "DataType3", as shown in Figure1. FunA Block can then read this value that is used to produce the desire. FunB instead, produces a wish for AGENT2, and through the KSAMANAGER can write this information in the state AGENT2. Since the state of a generic agent is visible and accessible to all other agents, KSA has been brought out of the agents. This allows for sharing also global objectives among agents. For example, AGENT1 in Figure 1 is part of a cooperative MAS composed of two agents as shown in Figure 2. The application of the representation by bigraph of the MAS allowed for recognizing and extracting the KSA as a node that contains the states of all agents.

With the following example it is possible to highlight the operation of the aforedescribed components (CORE, IOProcesing, INTENTIONS, KSAMANAGER, KSA, DataType), and the interaction between the various components of the proposed BDI model will be shown. As an example:

- Beliefs --> Sensor reading
- Desires --> Desires about the targets for the AGENT2 actuator (one of these by the AGENT1, subject to authorization thanks to the presence of the node in the KSAMANAGER as shown in Figure 2)
- Intentions --> the fusion between FunA and FunB. AGENT1 sees a sensor connected to it. The Reader node in IOProcessing manipulates the read data (e.g. voltage) in order to project it into the KSA (in the DataType3 column, where AG1Reading indicates the reading, for example 8-bit scaled, present in the AGENT1 line, from which AG1 is placed before the word reading).

Once this FunA (an "INTENTIONS") takes in input this reading (from KSA) and with its logic, produces a desire. This desire belongs to the DataType2 Column and it is contained in the row of the same agent, hence the name AG1DesireOutFunA. Since it is a desire, it will be taken into consideration by another agent (AGENT2). The other intention FunB, has no input (for reasons of simplicity and readability), therefore in some way, it deals with producing a desire for the AGENT2. This desire is written in the AGENT2 line in the DataType1 column (hence the edge with the name WriteAGENT2DataType1). AGENT2 operates as an actuator using FunC in conjunction with the assigned desire (the actuator operating mode, to operate in energy saving or not) of AGENT1. FunC takes AGENT2's wishes into account and calculates the setpoint to drive the actuator. The input for the actuator is provided by the drive node in the IOPprocessing. From what has been achieved with the bigraph application and the specific proposed application (CPS), the KSA can be implemented by a matrix that contains columns with homogeneous data, and rows that represent the data of each agent in the MAS.

Given the nature of this matrix, that represent the KSA, the system needs a protocol that allows for the exchange of data in (near) real time.

V. Case Study

The proposed system has been designed to handle the hydraulic aspects of the reactor behavior operating in an industrial wastewater plant. In this system a reactor must be filled with some mixtures made up of various liquid components. Each mixture is composed of a given set of

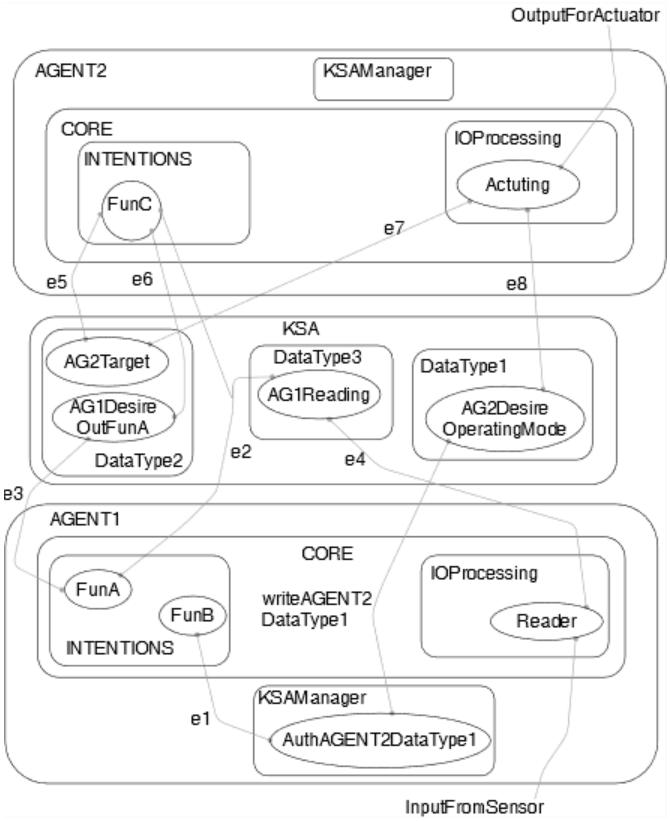

Fig. 2. Example of cooperative MAS with two Agents and unique KSA in which all environment information is represented and shared in real time.

liquids in prefixed percentages. The evaluated system is based on 22 units/agents, but for the sake of the clarity and without reducing the generality of the method, a simplified version comprising 5 devices is described below. Three units are equipped with a commercial controller (based on Toshiba's TB6560) for managing stepper motors and 3 peristaltic pumps. The pumps (dosers, with the possibility of micro-step) load a reactor specially designed with the liquids contained in three dispensers (e.g. three chemical reagents to be combined for a specific reaction according to assigned percentages).

A unit equipped with a PWM regulator controls a centrifugal pump used to empty the rector. A further unit reads the data from an ON/OFF level sensor.

These five units are part of a more complex system, developed thanks to the MAUI project.

Two goals have been defined in the system under evaluation:

- Local: for units with peristaltic pump: contributing to the mixture according to a known percentage; for units with centrifugal pump: empty the reactor; for units with level indicator: signal the achievement of the level.
- Global: filling the reactor up to the level of the sensor by mixing the three components according to assigned percentages.

A testbed based on ATmega328p processors board has been realized to evaluate the performance of the proposed approach in designing of a MAS. The system is configured as an intelligent cooperative MAS communicating through the I2C bus, native protocol for this class of processors. The system consists of 6 identical units. Each unit contains the processor, an I2C for bus extension (P82B715 Texas instrument) and various power circuits. They are equipped with 8 digital I/O ports and 4 ports with A/D converter. Each unit hosts an intelligent AGENT (Figure 3). The bridge is also made with a device that is identical to the others but with a USB port. The latter device is used to connect the I2C system bus with the external world represented by a PC equipped with MATLAB and behaves similarly with the LEADER described in [7]. These devices are used in many applications and they are very suitable for implementing CPS. The definition of the KSA requires attention because it manages the information shared among agents. It includes both the status of each agent and the management of the whole system priority policies (i.e. can an agent modify the goal of another agent?). A critical parameter to be considered in the design stage is the dimension of the KSA. Indeed, the greater is its size the greater will be the transmission time to share it among agents, but the greater will be also the awareness of each agent about the environment. Making each agent aware of its surrounding environment and defining some inference rules give to the MAS the ability to self-assess its possibility to meet global goals and identify the responsible of an eventual failure. Indeed, each agent implements a deductive process using some inference rules to analyze the compatibility between its local goal and the information into KSA. The agent that is not able to meet its local goal is the one handling the failure (probably a hardware failure). At this stage the malfunction can be treated according to appropriate procedures. Bigraphs have been used both to size the KSA and to define the reaction rules that allow to build the formal representation (i.e. the bigraph) of the entire system.

A MAS presents many challenges in terms of intelligent

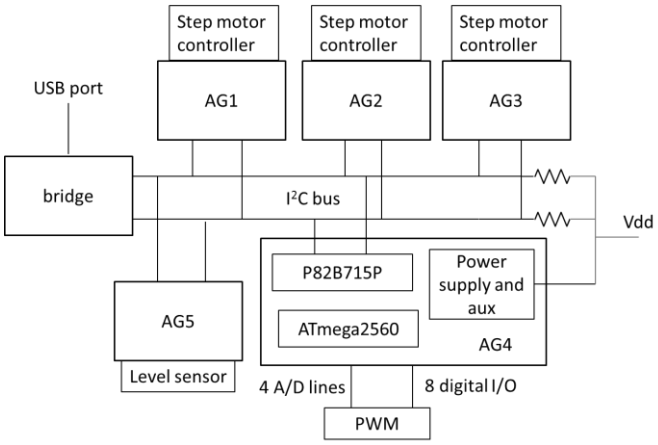

Fig. 3. Scheme of the architecture being tested. The system consists of 6 identical units. Each unit is connected through its I/O lines to specific devices dedicated to the interface with the real world

decision algorithms support. A distributed algorithm is often more practical and robust than a centralized algorithm. Agents are in general characterized by strong autonomy. This means that an agent operates without the direct intervention of some other entities and has control over their actions.

Assigning the goals to an agent means that actions on environment should be done to achieve some specified desires and that the agents show a sort of rational behavior in the environment. The term "behavior" usually refers to the action that is performed after receiving a set of inputs from sensors. Agents in a distributed decision-making architecture are often self-interested, i.e. they optimize decisions according to local conditions, with limited consideration of overall performance and constraints. Often agents operate in competitive structures, but structures with cooperating agents are currently gaining interest [6]. These structures are characterized by multiple objectives and the possibility of achieving even part of the objectives assigned to the individual agent.

The need to manipulate heterogeneous data or data characterized by different ontologies should be considered

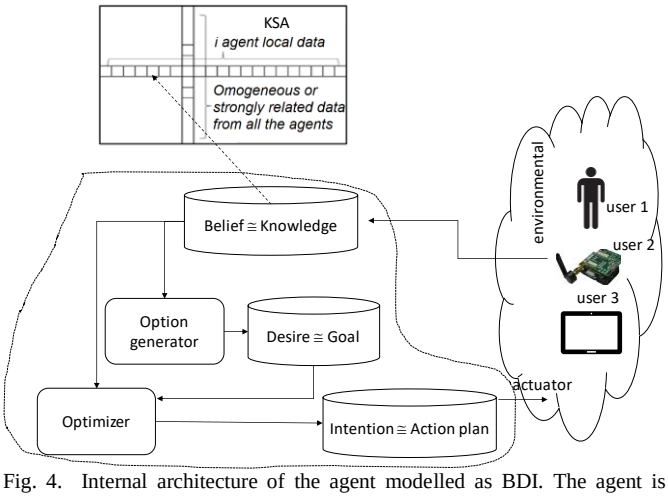

Fig. 4. Internal architecture of the agent modelled as BDI. The agent is connected to the external environment through sensors and actuators.

while designing a cooperating MAS. In the proposed system, agents, operating in a reactive MAS, must follow specific policies at local level respecting the limits imposed at global level. Communication among agents is therefore relevant for the implementation of these policies and relative algorithms. Moreover, global and local policies must have the same limits, so that the set, related to the latter, is incorporated in the set related to the former.

From an architectural point of view, an agent operating in a MAS needs to expand its knowledge of the domain in which the system operates. The visibility must be such as to allow the agent to see all the variables related to both local and global and/or shared objectives. As explained above, using also bigraphs in designing stage of the MAS allowed for isolating an entity identified as KSA matrix. This entity is responsible for sharing all the knowledge that characterizes the MAS. In KSA matrix each row holds information about the state of each agent (both sensors and actuators), along with their local actions, while each column represents a specific data parameter (e.g., the temperature for each room). In Figure 4 an example of KSA matrix containing the data for a single agent is presented. In this study a simple communication protocol, based on the I2C serial bus, has been developed to assess the efficiency of the proposed approach.

It is of fundamental importance that the global status of the system is shared by all the units to pursue common objectives. For the prototyping phase it was decided to use the TWI (two wire interface) processor port (the well-known I2C port of the Arduino board).

The master-slave communication model has been adopted. The master, identified by a simple token model among the units involved in the MAS, has the task of transmitting the KSA matrix without generating overwriting with data loss. In a very simplified way, if N is the total number of units connected to the bus; the token is assigned to the m-th unit that acts as a master. The master asks the i-th slave (i=0, ..., N) m≠ i, for its status vector and then transmits it to the other N-1 slaves in a sequential way. This update strategy allows for avoiding the expired data in the matrix. Indeed, the i-th slave is the only unit that can modify its state vector. If the matrix were instead completely updated for each cycle, the i-th slave would also see its state overwritten with the possibility of deleting more recent data. Tests have been carried out both with a physical master and with a dynamic master managed by a special token. The communication protocol was tested at a clock frequency (f) of 400 KHz [9]. Each message exchanged by means of the protocol requires few clock cycles to compute. In the proposed system (composed by N units each one with M local data), the protocol requires a time estimated by:

$$T=(N^2*(11+M*9)+N*34)/f \quad (1)$$

VI. Experiments and Results

In this section, two aspects of the proposed system have been evaluated and reported: the construction of the bigraph for the proposed MAS with the BigMC tool and the results obtained using this system in various real cases.

A. Methodological results

The CPS with MAS architecture described in the previous section has been successfully designed using the bigraph approach.

To construct the bigraph model according to the structure described above, a multiscale approach is adopted as shown in [31].

The scale options of our approach are sorted as follows:

- Scale "Environment", which represents the topological entities of environments such as a room or container;
- Scale "Objects", which represents all passive entities, which are unable to perform any action without interaction/possession of agents, such as sensors, actuators, transmission media, computers or humans;
- Scale “Subjects”, which represents the structure of an agent, as shown above;
- Scale "KSA", which represents the nodes of beliefs and desires of the agents;
- Scale "Software Components", which are the software components in the structure of each agent, e.g. FunA, FunB, Reader and AuthAGENT2DataType1 in Figure 1, with the interconnection between the components of the Agent software and the objects present in the environment, and also with interconnection between the components of the agent software and the node in KSA; the construction of the bigraph is done with the BigMC tool, and the code in the case study shows how the multi-scale approach is adopted.

Thereafter, the relevant BigMC code of the proposed multi-scale approach, that is adopted to obtain the formal description of the system through a bigraph is reported. The BigMC terms language and the structure of a file declaring a bigraph can be seen in [10] while next some statements of the BigMC file describing the proposed MAS are shown to explain how the multi-scale approach works.

The nodes and links declarations are omitted since deductible from reported code, but also because they are not relevant to explain the multi-scale approach. As said above, the proposed multi-scale approach is composed of 5 scales and the declaration code of each one is presented below:

- the Scale Environment is declared through a bigraph composed of one node called Room, which represents the room where our devices are placed;
- The passive entities belonging to Scale Objects, which are unable to perform any actions without interaction/possession of agents, are nested in the Scale Environment by the following statement:

```
Room ->
Room.(Reactor.SensorLevel[OUTSensorLevel] |
      ComputerAG1 | ComputerAG2 | ComputerAG3 | ComputerAG4 |
      ComputerAG5 |
      Human |
      PumpInj1[INNActuating1,OUTsteps1] |
      PumpInj2[INNActuating2,OUTsteps2] |
      PumpInj3[INNActuating3,OUTsteps3] |
      PumpDrain[INNActuating4,OUTsteps4] |
      ComChannel
    );
```

In the square brackets there are the inner/outer links of these entities, denoted by INN/OUT prefix.

- The active entities belonging to Scale Subjects, which represent the structure of an agent, are nested in the Scale Environment by the following statement:

```
Human ->
Human.AgentHuman[OUTSetAG1mixtureRatio,
OUTSetAG2mixtureRatio,
                 OUTSetAG3mixtureRatio];
ComputerAG1 -> ComputerAG1.Agent;
ComputerAG2 -> ComputerAG2.Agent;
ComputerAG3 -> ComputerAG3.Agent;
ComputerAG4 -> ComputerAG4.Agent;
ComputerAG5 -> ComputerAG5.Agent;

Agent ->
Agent.(CORE.(Intentions | IOmanager)
                | KSA Manager
      ) ;
```

- The Scale KSA, which represents the nodes of beliefs and desires in KSA, i.e. the matrix elements, is nested in the communication channel as follows:

```
ComChannel -> ComChannel. KSA.
  (MixtureRatio1.(AG1mixtureRatio[INNAG1mixtureRat
io,
                        OUTAG1mixtureRatio]
      |AG2mixtureRatio[INNAG2mixtureRatio,
                        OUTAG2mixtureRatio]
       |AG3mixtureRatio[INNAG3mixtureRatio,
                        OUTAG3mixtureRatio])
        |nStepFill2.(AG1NstepsFill[INNAG1NstepsFill,
                        OUTAG1NstepsFill]
          |AG2NstepsFill[INNAG2NstepsFill,
                        OUTAG2NstepsFill]
           |AG3NstepsFill[INNAG3NstepsFill,
                        OUTAG3NstepsFill])
      |OnOffSensLev3.(AG5statusSensor[
                        INNAG5statusSensor,
                        OUTAG5statusSensor])
   |nStepDone4.(AG1StepsDone[INNAG1StepsDone,
                        OUTAG1StepsDone]
         |AG2StepsDone[INNAG2StepsDone,
                        OUTAG2StepsDone]
          |AG3StepsDone[INNAG3StepsDone,
                        OUTAG3StepsDone]
          |AG4StepsDone[INNAG4StepsDone,
                        OUTAG4StepsDone])
  |OnOffActuatingPump5.(
          AG1ActuatingPump[INNAG1ActuatingPump,
                        OUTAG1ActuatingPump]
    |AG2ActuatingPump[INNAG2ActuatingPump,
                        OUTAG2ActuatingPump]
      |AG3ActuatingPump[INNAG3ActuatingPump,
                        OUTAG3ActuatingPump]
      |AG4ActuatingPump[INNAG4ActuatingPump,
                        OUTAG4ActuatingPump])
  );
```

This statement represents the KSA matrix of the CPS. In this matrix, MixtureRatio1 is the first column, AG1mixtureRatio is the element in the first column-first row, AG2mixtureRatio is the element in the first column-second row, and so on.

- The entities belonging to Scale Software Components, are nested in the Agent by the following statement:

```
ComputerAG1.Agent.(CORE.(Intentions | IOmanager) |
$1)->
ComputerAG1.Agent.(CORE.(
     Intentions.(
          SetNstepsFill[OUTAG5statusSensor,
                        OUTAG1StepsDone,
                        INNAG1NstepsFill]
        |SetResetStepsDone[OUTAG1ActuatingPump,
                        INNAG1StepsDone]
        |SetActuatingPump[ OUTmAG1StepsDone,
                        INNAG1ActuatingPump]
     )
     |IOmanager.(
          SetMixtureRatio[
                        OUTSetAG1mixtureRatio,
                        OUTAG5statusSensor,
                        OUTAG1NstepsFill,
                        OUTAG1StepsDone,
                        OUTAG1mixtureRatio,
                        INNAG1mixtureRatio]
        |ReaderStepsDone[ OUTsteps1,
                        INNAG1StepsDone]
        |ActuatingPump[INNActuating1,
                        OUTAG1mixtureRatio]
         )
       )
      |$1);
```

Where the wildcard $1 means “all nodes and links placed in Agent node except CORE node and its contents” (e.g. the remaining KSAMANAGER node). The remaining agents’ declarations follow the same structure. The BigMC file ends with %check, that provides, in the last state generated in output, the complex bigraph representative of our system scenario.

The bigraph obtained by means of application of reaction rules is presented hereinafter. The full output is too large to be integrated in the paper, so a simplified view is shown.

```
Welcome to BigMC!
> /usr/local/bigmc/bin/bigmc -m 10000 -r 50 -p
/tmp/bigmc_model7918810766099976581.bgm
1:Room.nil
2:Room.(Reactor.SensorLevel[OUTSensorLevel].nil
| ComputerAG1.nil
| ComputerAG2.nil
| ComputerAG3.nil
| ComputerAG4.nil
| ComputerAG5.nil
| Human.nil
| PumpInj1[INNActuating1,OUTsteps1].nil
| PumpInj2[INNActuating2,OUTsteps2].nil
```

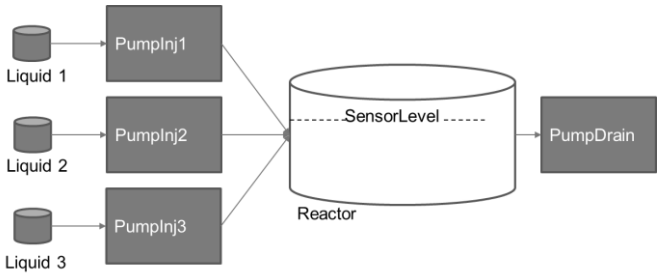

Fig. 5. Test-bed setup related to experiments conducted on the MAS. The figure is like Figure 3, but in this case it highlights the physical components. The testbed is based on a reactor where a liquid mixture must be realized

```
| PumpInj3[INNActuating3,OUTsteps3].nil
| PumpDrain[INNActuating4,OUTsteps4].nil
| ComChannel.nil)
.
.
.
4097: Room.(ComputerAG4.Agent.(KSA Manager.nil
        | CORE.(IOmanager.(

ActuatingPump[OUTSensorLevel,
INNActuating4].nil
                | ReaderStepsDone[OUTsteps4,

INNAG4StepsDone].nil
          )
     |
Intentions.(SetActuatingPump[OUTAG5statusSensor,

INNAG4ActuatingPump].nil
              |
SetResetStepsDone[OUTAG4ActuatingPump,
                        INNAG4StepsDone].nil
    )
  )
) | ComputerAG1.Agent.(...
[mc::step] Complete!
[mc::report] [q: 0 / g: 4097] @ 4098
```

B. Real case experiments

According to the previous sections and as a result of the study of the system with the bigraphs, the matrix representing the KSA was designed. This data structure is shared among all the units that can "see" the environment in which they operate. Table I reports the KSA for the part of interest of the presented test.

A testbed has been realized according to Figure 5. The software is characterized by multiple procedures executed in a loop section, and specifically: a manager of the KSA synchronization and propagation (Table I), a set of local reactive systems able to interact with external/internal conditions to each user agent [24], [32]. Several experiments were conducted to evaluate the MAS performance and its capability to react to different scenarios. The system has been tested using more than ten different mixtures simulating various negative operative conditions (such as fault of one or more pumps, one or more exhaust components, the presence of foam in the reactor (in this case the sensor level gives wrong mixture levels, etc.). In each experiment the system behavior has been studied analyzing the three systems' working phases: learning phase, operative phase without error and operative phase with exception revealed.

TABLE I
KSA FOR THE MAS MODELLED

	Mixture ratio	nStep Fill	On/Off SensLev	nStep Done	On/Off Actuation Pump
AG1 (Pumpinj1)	value	value		value	value
AG2 (Pumpinj2)	value	value		value	value
AG3 (Pumpinj3)	value	value		value	value
AG4 (Pumpdrain)					value
AG5 (SensorLevel)			value		

In each row the data relate to each agent. The bridge can write the values that represent the goals in the lines of the other agents. nSpepDone = incremental pheristaltic pump turn counter; OnOffActuatingPump = pump status 0 → off, 1→ on; MixtureRatio = mixture percentage for the single reagent; OnOffSensLev = level sensor status 0→ off, 1→ on (reactor empty); nStepFill = turns of the peristaltic pump to fill the reactor.

Figure 6 shows an example of such evaluations with a graphical view of the three scenarios analyzed during one of the tests.

- Scenario 1: Learning phase – The first part of each experiment is based on a learning phase where each agent operates in order to learn how to work in cooperation for the global goal achievement. Starting from the consideration that any stepper pump has a constant flow, in this phase each user agent related to a Stepper Injection Pump learns how many steps are necessary to fill completely the reactor so as to contribute with its specific ratio. After any filling the Pump Drain operates to empty the reactor.
- Scenario 2: Operating phase with no error – In this phase the whole set of agents starts to cooperate in order to achieve the global goal. In this scenario the whole system works perfectly, and each user agent does not observe any errors. In fact, three perfect cycles of filling and emptying are realized (Figure 6).
- Scenario 3: Operating phase with exception revealed. During the last cycle reported in Figure 6 (see enlargement), the liquid 2 tank becomes empty. This means that the user agent related to PumpInj2 cannot cooperate to reach global goals. Each agent can observe that an anomaly occurs (for instance the user agent related to Pumpinj1 will observe that a bigger number of steps is required to fill the reactor) and the whole system reacts by defining an alert condition related to the defined global goals. In this experiment the reactive system acquires knowledge about the fact that the global goal is impossible to be achieved with the tank 2 in empty condition.

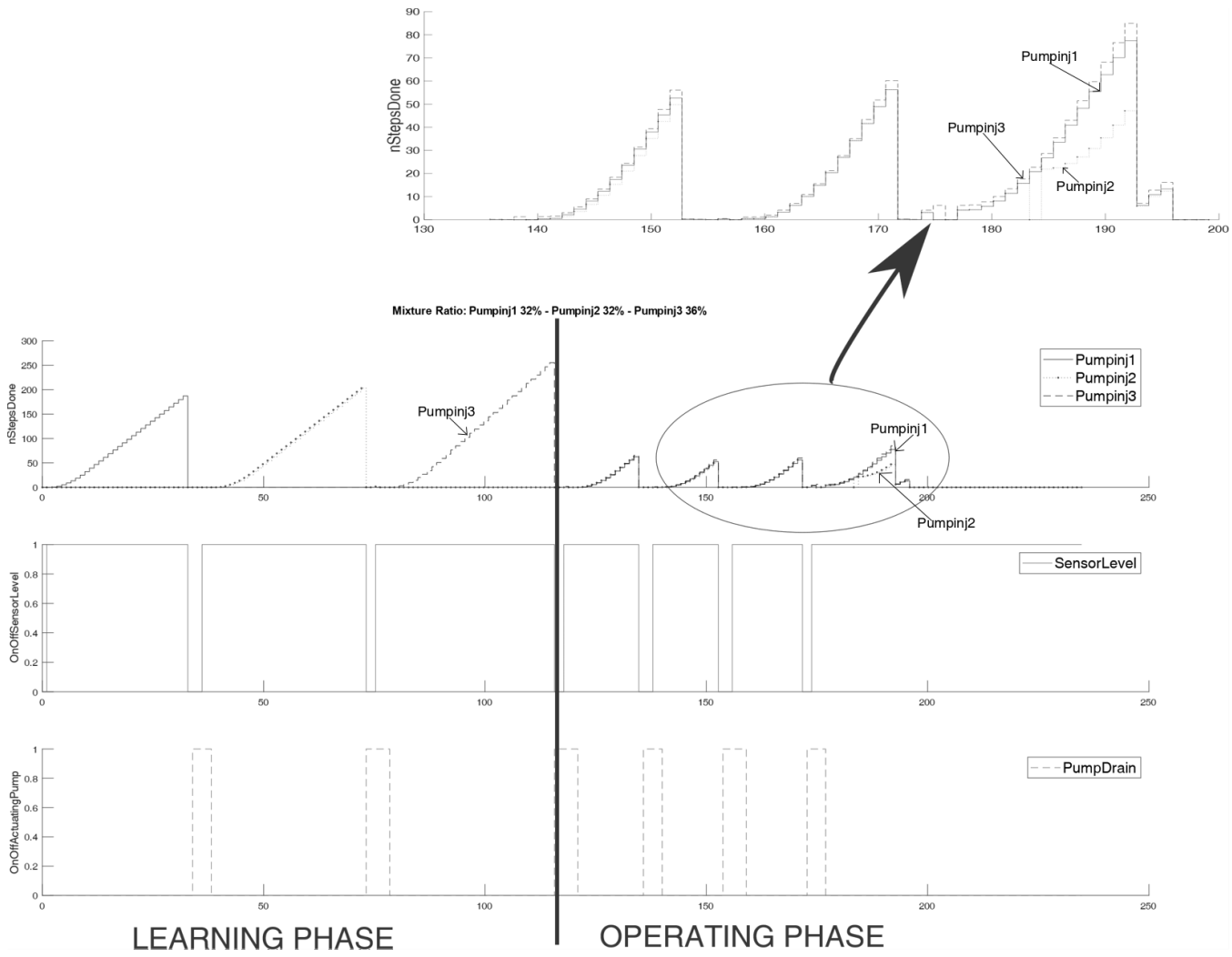

Fig. 6. Experimental results related to tests conducted. The experimental setup was based on 3 stepper motor pumps for liquid injection in a common reactor and 1 on/off pump for actuating the reactor emptying.

VII. Conclusions

This paper proposes a model for designing and implementing a CPS based on cooperative MAS paradigm. The identification of this model requires the use of design tools and a system architecture that are able to represent and manage the characteristic aspects of the system under analysis such as: topology, communication, the objectives of the entities present in the system (local goals for each agent) and cooperation to meet objectives of global interest (global goals of the MAS).

The proposed approach sees a CPS as a multi-scale problem resolvable using a cooperative MAS. The work has been characterized by two parallel flows. The first one, starting from the analysis of the state of the art of the techniques used in designing this class of system, has led to the identification of Bigraph and BRS as tools capable of fully representing the concepts of location, connectivity and agent. The second workflow, starting from the analysis of the state of the art of the paradigms used to model and implement the various entities in the layer in close contact with the physical process of the CPS, has led to the introduction of a specific entity (KSA) representing the environment in which all the cooperative agents operate.

This entity has been implemented using the characteristic loop of the used hardware development environment (ATMega 328p processor or Arduino type boards). The loop first provides the procedures for replication of KSA (virtual) in all agents (physical) and then the computational procedures related to each individual agent. The MAS and its KSA has been defined through the Belief-Desires-Intentions (BDI) model. The proposed system has been used in an application case of the MAUI project that is a project funded by the Lombardy Region (Italy) and aimed to identify innovative monitoring techniques for the control of industrial wastewater. In all the experiments the system has shown an intelligent and proactive behavior changing its goals to handle the different external conditions in which it operates.

This work shows that using the proposed multi-scale model, a cooperative MAS and the notion of KSA, it is possible to proceed to the formal design of a CPS with interesting features such as the ability to meet global goals and self-diagnostic of failures.

These results have also been confirmed by the BigMC tool, which produces the formal model of the system as output. For what concerns the implementation phase, a protocol capable of representing and manipulating KSA has been used. Through this KSA each agent in the MAS has fully visibility of the environment in which it operates. This

allows each agent to cooperate in order to achieve global objectives starting from its individual local objectives.

Future works will focus on the adoption of the bigraph during the run-time execution of the system. The goal is to develop a decision support system (DSS). Using a Java LibBig library, it is possible to use its model checker for a reachability analysis. By observing the flow of the KSA matrix, the application can update the bigraph (for reaction rules) and, by formalizing the situation of interest in terms of applicability of reaction rules, the checker can detect when this will be done, and then the system will take appropriate measures to address it.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," *Futur. Gener. Comput. Syst.*, vol. 25, pp. 599–616, 2009.

[3] H. S. Nwana, "Software agents: an overview," *Knowl. Eng. Rev.*, vol. 11, no. 03, p. 205, Sep. 1996.

[4] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," *Knowl. Eng. Rev.*, vol. 10, no. 02, p. 115, Jun. 1995.

[5] R. Capilla, F. Nava, S. Pérez, and J. Dueñas, "A web-based tool for managing architectural design decisions," *ACM SIGSOFT Softw. Eng. Notes*, vol. 31, 2006.

[6] Y. Hong, G. Chen, and L. Bushnell, "Distributed observers design for leader-following control of multi-agent networks," *Automatica*, vol. 44, no. 3, pp. 846–850, Mar. 2008.

[7] Q. Shen, B. Jiang, P. Shi, and J. Zhao, "Cooperative Adaptive Fuzzy Tracking Control for Networked Unknown Nonlinear Multiagent Systems With Time-Varying Actuator Faults," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 3, pp. 494–504, Jun. 2014.

[8] K. Zafar, S. Qazi, and A. Baig, "Mine Detection and Route Planning in Military Warfare using Multi Agent System," in *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, 2006, pp. 327–332.

[9] V. Di Lecce, A. Amato, and M. Calabrese, "Data integration in distributed medical information systems," in *2008 Canadian Conference on Electrical and Computer Engineering*, 2008, pp. 001497–001502.

[10] A. T. E. Dib and Z. Sahnoun, "Model Checking of Multi Agent System Architectures Using BigMC," 2015, pp. 1717–1722.

[11] J. C. F. Li, M. Lei, and F. Gao, "Device-to-device (D2D) communication in MU-MIMO cellular networks," in *2012 IEEE Global Communications Conference (GLOBECOM)*, 2012, pp. 3583–3587.

[12] B. Manoj, R. Rao, and M. Zorzi, "CogNet: a cognitive complete knowledge network system," *IEEE Wirel. Commun.*, vol. 15, no. 6, pp. 81–88, Dec. 2008.

[13] A. Amato, V. Di Lecce, C. Pasquale, and V. Piuri, "'Web agents' in an environmental monitoring system," in *CIMSA. 2005 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, 2005.*, pp. 262–265.

[14] A. Jansen and J. Bosch, "Software Architecture as a Set of Architectural Design Decisions," in *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*, pp. 109–120.

[15] G. Marquez and H. Astudillo, "Selecting components assemblies from non-functional requirements through tactics and scenarios," in *2016 35th International Conference of the Chilean Computer Science Society (SCCC)*, 2016, pp. 1–11.

[16] A. Jansen, J. Der Ven, P. Avgeriou, and D. Hammer, "Tool Support for Architectural Decisions," in *2007 Working IEEE/IFIP Conference on Software Architecture (WICSA'07)*, 2007, pp. 4–4.

[17] L. Yu, W. T. Tsai, Y. Jiang, and J. Gao, "Generating Test Cases for Context-Aware Applications Using Bigraphs," in *2014 Eighth International Conference on Software Security and Reliability*, 2014, pp. 137–146.

[18] MathWorks, "Simulink." [Online]. Available: http://www.mathworks.com/products/simulink/index.html.

[19] G. Holzmann, *Spin Model Checker, the: Primer and Reference Manual*, First. Addison-Wesley Professional, 2003.

[20] A. Louati, K. Barkaoui, and C. Jerad, "Temporal Properties Verification of Real-Time Systems Using UML/MARTE/OCL-RT," 2015, pp. 133–147.

[21] G. Cattani, J. J. Leifer, and R. Milner, "Contexts and embeddings for a class of action graphs," 2000.

[22] T. J. Koo, B. Sinopoli, A. Sangiovanni-Vincentelli, and S. Sastry, "A formal approach to reactive system design: unmanned aerial vehicle flight management system design example," in *Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design (Cat. No.99TH8404)*, pp. 522–527.

[23] P. Sewell, "From rewrite rules to bisimulation congruences," *Theor. Comput. Sci.*, vol. 274, no. 1–2, pp. 183–230, Mar. 2002.

[24] C. Tsigkanos, L. Pasquale, C. Ghezzi, and B. Nuseibeh, "On the Interplay Between Cyber and Physical Spaces for Adaptive Security," *IEEE Trans. Dependable Secur. Comput.*, vol. 15, no. 3, pp. 466–480, May 2018.

[25] P. Melidis, E. Vaiopoulou, and A. Aivasidis, "Development and implementation of microbial sensors for efficient process control in wastewater treatment plants," *Bioprocess Biosyst. Eng.*, vol. 31, no. 3, pp. 277–282, Apr. 2008.

[26] G. Mannina, G. Freni, G. Viviani, S. Sægrov, and L. S. Hafskjold, "Integrated urban water modelling with uncertainty analysis," *Water Sci. Technol.*, vol. 54, no. 6–7, pp. 379–386, Sep. 2006.

[27] G. Freni, G. Mannina, and G. Viviani, "Uncertainty in urban stormwater quality modelling: The influence of likelihood measure formulation in the GLUE methodology," *Sci. Total Environ.*, vol. 408, no. 1, pp. 138–145, Dec. 2009.

[28] H. Ceballos and R. Brena, "Finding Compromises Between Local and Global Ontology Querying in Multiagent Systems," 2004, pp. 999–1011.

[29] A. Mansutti, M. Miculan, and M. Peressotti, "Multi-agent Systems Design and Prototyping with Bigraphical Reactive Systems," 2014, pp. 201–208.

[30] A. T. E. Dib and Z. Sahnoun, "Formal Specification of Multi-Agent System Architecture," *CEUR Workshop Proceedings*, vol. 1294. 2014.

[31] A. Gassara, I. B. Rodriguez, M. Jmaiel, and K. Drira, "A formal method for modeling deployment architectures based on bigraphs," *ACM SIGAPP Appl. Comput. Rev.*, vol. 15, no. 2, pp. 8–16, Aug. 2015.

[32] H. Zhang, F. L. Lewis, and Z. Qu, "Lyapunov, Adaptive, and Optimal Design Techniques for Cooperative Systems on Directed Communication Graphs," *IEEE Trans. Ind. Electron.*, vol. 59, no. 7, pp. 3026–3041, Jul. 2012.

Vincenzo Di Lecce received the doctoral degree in electric engineering, cun laude with honors, from the University of Bari in 1980. After 2 years in industry, he won a Selenia Spazio Italiana fellowship for research on radar signal elaboration. In 1986 he became technical chief of the electronic calculator laboratories at the Engineering Faculty of the Politecnico of Bari and at present, he is professor of computer science there. His research activities include Artificial Intelligence and data analysis, environmental application of AI and intelligent sensor. He has published about 200 science papers in these fields.

Alberto Amato was born in Taranto, Italy, in 1976. He received the B.Sc. degree in environmental engineering from the Politecnico di Bari, Bari, Italy, in 2003 and he obtained the Ph.D. in Information Technology from Università degli Studi di Milano, Milan, Italy in 2010. He is the author or coauthor of about 50 papers. His current research interests include clustering algorithms, image retrieval, video analisys, sensor network and knowledge-based systems.

Alessandro Quarto (M'19) was born in Taranto, Italy in 1981.He received the master's degree in Information Engineering from the Polytechnic of Bari, Italy in 2007. Currently is CEO of an innovative SME. His main research interests are on Artificial Intelligence, Multi Agents Systems and IoT.

Marco Minoia was born in Gioia del Colle, Bari, Italy, in 1993. He received the bachelor's degree in information and automation engineering from Polythecnic of Bari, Italy, in 20016. He is currently pursuing the master's degree in computer engineering at Polythecnic of Bari, Italy. In 2018, he starts a research activity with AeFLab at Polythecnic of Bari. His research interest includes Multi-Agent Systems and Cyber-Physical Systems.