Self-organized Cyber-Physical Conveyor System using Multi-agent Systems

Paulo Leitão[1], José Barbosa[1], Gustavo Funchal[1,2] and Victoria Melo[1,2]

[1]Research Centre in Digitalization and Intelligent Robotics (CeDRI)
Instituto Politécnico de Bragança
Campus de Santa Apolónia, 5300-253 Bragança, Portugal
{pleitao, jbarbosa}@ipb.pt
[2]Federal University of Technology – Parana (UTFPR)
Campus de Cornélio Procópio, PR, Brasil
{gustavofunchall, victoriac.a.melo25}@gmail.com

Abstract

The adoption of industrial cyber-physical systems is facing several challenges, with artificial intelligence and self-organization techniques assuming critical aspects to be considered in the deployment of such solutions to support the dynamic evolution and adaptation to condition changes. This paper describes the implementation of a modular, flexible and self-organized cyber-physical conveyor system build up with different individual modular and intelligent transfer modules. For this purpose, multi-agent systems are used to distribute intelligence among transfer modules supporting pluggability and modularity, complemented with self-organization capabilities to achieve a truly self-reconfigurable system. Furthermore, Internet of Things and Artificial Intelligence technologies are used to enable the real-time monitoring of the system, aiming the detection and prevention of anomalies in advance, and to enable the protection from possible external threats.

Keywords: Multi-agent systems, Cyber-physical systems, Self-organization, Internet of Things, Machine learning.

Computing Classification System: Computing methodologies - Artificial intelligence - Distributed artificial intelligence - Multi-agent systems

1 Introduction

The 4th industrial revolution, also known to Industry 4.0, is characterized by the digitization of traditional factories to allow the companies to be more competitive facing the strong pressures imposed by global markets and demanding customers, reflected in terms of customization, quality and diversity of products, and the fast response to the demand. Cyber-physical systems (CPS) (Leitão, Karnouskos and Colombo, 2016; Lee, 2008) constitute the backbone to implement the Industry 4.0 principles (Kagermann, Wahlster and Helbig, 2013), enabling the next generation of production systems, that are characterized by the integration, in a network and large scale manner, of several heterogeneous, connected and smart processes, systems

and devices, combining the cyber and physical counterparts, which allows improving the production systems' performance, responsiveness and reconfigurability, while efficiently managing the complexity and uncertainty. The realization of such CPS systems involve the application of several emergent and disruptive technologies, such as Internet of Things (IoT), cloud and edge computing, Artificial Intelligence (AI), Big data analytics, collaborative robotics, virtual reality and cybersecurity.

Multi-agent systems (MAS) (Wooldridge, 2002; Leitão, 2009) is a suitable approach to realize such large-scale and complex industrial CPS, due to its inherent characteristics of modularity, autonomy, decentralization and adaptation to emergence without external intervention. In particular, MAS decentralizes the control functions over distributed and intelligent software agents, running in cloud and edge computational layers, that cooperate together to achieve the system goals facing the condition change and reconfiguration needs. To tackle these issues, such agent-based CPS solutions need to exhibit several capabilities, such as reactiveness, robustness and self-organization, supporting the reconfiguration, adaptation and evolution in a very fast, automatic and self-manner (Barbosa, Leitão and Teixeira, 2017).

However, the application of MAS, by itself, does not completely guarantee the achievement of truly reconfigurable systems, being necessary to combine them with emergency and self-organization mechanisms to support the dynamic structure re-configuration. If emergency can be achieved due to the inherent capabilities of using the MAS principles, the use of self-organization can be addressed by enhancing these systems with simple but powerful bio-inspired mechanisms. Particularly, it is advised to understand the self-organization mechanisms working in biology and nature by millions of years and translate them, case-by-case, to engineer the new cyber-physical production systems. As stated by (Leitão et al., 2016), the use of bio-inspired mechanisms, and particularly self-organization, is one main challenge for the adoption of CPS in industrial environments, being its industrial adoption far from the expectation.

Having this in mind, this paper aims to discuss the benefits of using agent-based systems enhanced with self-organization capabilities to enable the implementation of modular and reconfigurable industrial CPS, aligned with the Industry 4.0 principles. For this purpose, a modular and self-organized CPS solution, developed using the MAS technology and enhanced with AI and self-organization techniques, is deployed to create a modular conveyor transfer system build-up of several individual Fischertechniks conveyors. The agents were running at edge computing layer, deployed in Raspberry Pi single board computers, allowing the automatic, dynamic and on-the-fly pluggability and system reconfiguration, i.e. without the need to stop, re-program and re-start the system, and thus dealing more effectively with unpredicted behaviour.

The rest of the paper is organized as follows. Section 2 overviews the related work related to use MAS and self-organization to implement complex, scalable, robust and reconfigurable CPS in industrial environments. Section 3 presents the case study based on modular Fischertechniks conveyors, highlighting the main components and their functionalities. Section 4 describes the implementation of the modular cyber-physical transport system using the MAS technology embedded with some AI techniques. Section 5 details the enhancement of the

agent-based system with self-organization capabilities to face the on-the-fly reconfiguration, and finally, Section 6 rounds up the paper with the conclusions.

2 Related Work

CPS is an emergent approach that focuses on the integration of computational applications with physical devices, being designed as a network of interacting cyber and physical counterparts to form a large system (Leitão et al., 2016), being applied to different fields, namely manufacturing, energy, health and building automation. According to (Leitão et al., 2016), the adoption of CPS in industrial environments is faced by several challenges, being the use of AI techniques and bio-inspired mechanisms, and particularly self-organization, promising aspects to be considered in the deployment of such CPS solutions to support the dynamic evolution and adaptation to condition changes. Self-organization is a way to achieve the adaptation to the dynamic evolution of the environment, being possible to be defined as *the ability of an entity/system to adapt dynamically its behaviour to external conditions by re-organizing its structure through the modification of the relationships among entities without external intervention and driven by local and global forces* (Leitão, 2009).

In spite of the system re-organization may also occur in structures with centralized control, the entire potential of self-organization occurs in distributed system architectures, as MAS are, which do not follow a rigid and predictable structure organization (Bousbia and Trentesaux, 2002). In this case, each individual entity is regulated according to a self-organized behaviour and the overall self-organized system emerges from the non-linear interactions among individuals, involving amplification and cooperation, and the sensitivity to initial conditions (i.e. the butterfly effect). The resulting behaviour is consequently more nervous and difficult to predict, being required the existence of regulation mechanisms to ensure that expected behaviours will actually emerge, and to maintain the system in a stable state.

This concept can be found in several domains, such as biology (e.g., the ants foraging and birds flocking), physics (e.g., the 2nd thermodynamics law) and social organization (e.g., traffic in crowded environments). In manufacturing domain, bio-inspired self-organized mechanisms can be applied to support the dynamic and on-the-fly re-configuration of the control structure, facing the agile reaction to unpredictable condition changes. Several applications combining self-organization capabilities with MAS in manufacturing domain can be found in literature, namely, P2000+ (Bussmann, Schild and Baumgaertel, 2000), ADACOR (Leitão and Restivo, 2006), PROSA + ants (Valckenaers, Hadeli, Kollingbaum, Brussel and Bochmann, 2002), AirLiquide (Miller, 2007), AIP-PRIMECA (Zambrano, Bonte, Prabhu and Trentesaux, 2014) and ADACOR2 (Barbosa, Leitão, Adam and Trentesaux, 2015).

These approaches are mainly running in laboratorial environments, being noticed a reduced application in industrial environments (note that weak self-organization implementations, that are those following centralized structures, are not considered in this work). In fact, stakeholders are still afraid to use these emergent concepts, being required their exposition to evidences of the real benefits of using self-organization in large and complex industrial CPS. However, the recent availability of powerful and low-cost single-board computer platforms with IoT con-

nectivity, e.g., Raspberry Pi, can contribute for the deployment of MAS solutions embedding AI and self-organization algorithms at the edge computational layer, and thus truly developing industrial CPS solutions that face industrial requirements.

3 Case Study

The problem considered in this work is related to a conveyor transfer system comprising a sequence of modular Fischetechnik conveyors, each one providing the functionality to convey parts from the input to the output position. Each individual conveyor comprises a belt operated by a 24V DC motor and 2 light sensors to detect parts in the input and output positions, also operating at 24V, as illustrated in Figure 1.

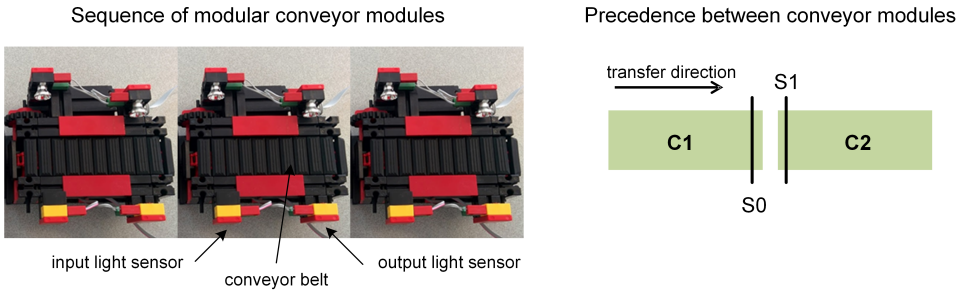

Figure 1: Details of the modular conveyor transfer system.

The parts are transferred among the conveyor modules from the input location of the first module to the output location of the last module. The transfer between two consecutive transfer modules is performed in the following manner (see Figure 1): i) the conveyor C2 starts its motor when the part arrives to sensor S0, and ii) the conveyor C1 only stops its motor when the part arrives to the sensor S1. This modus operanti establishes interdependencies between transfer modules that increases the complexity of the problem since the sequence of the transfer modules should be known when the logical control application is developed. A pertinent question that arises is related to the way the logical control of such modular and reconfigurable transfer system can be implemented.

The traditional approach uses a centralized logical control, e.g., based on an IEC 61131-3 program running in a PLC (Programmable Logic Controller). This solution is simple to program and is industrial adopted, but presents limitations in terms of system scalability and reconfigurability. In fact, the re-configuration of the conveyor system, e.g., adding a new conveyor or switching the order of the conveyors, is complex and time-consuming due to the existing interdependencies among conveyors. Another approach that could be considered is the use of the IEC 61499 standard (Vyatkin, 2011), that defines a generic model for distributed automation control systems but does not completely support the intelligent and autonomous decisions needed to implement the re-configuration on-the-fly.

Having this in mind, an alternative design approach to support the easy and on-the-fly reconfiguration of the conveyor system is to consider MAS technology to create a CPS based on several modular cyber-physical devices, where a software agent plays the role of the cyber part

controlling the physical conveyor device. This solution is only complete to reach the on-the-fly pluggability and reconfigurability if the agents are enhanced with AI and self-organization techniques.

4 Multi-agent Systems to Realize the Cyber-Physical Conveyor System

The proposed approach uses MAS technology embedding self-organization capabilities to implement a modular and reconfigurable conveyor transfer system. This section details the design and deployment of the agent-based CPS, which comprises the development of the individual modular cyber-physical conveyor devices and the design of the interactions that will allow the emergence of the overall CPS.

4.1 Building Modular Cyber-Physical Conveyor Devices

The proposed cyber-physical conveyor system comprises several modular cyber-physical conveyor devices, each one performing its own operation and interacting with the others to ensure the emergence of the transfer functionality of the entire conveyor system. In this way, an individual cyber-physical conveyor device comprises two symbiotic parts: the cyber part, provided by the processing capabilities offered by the agents running in a single board computer and controlling the behaviour of the physical part, that is constituted by the hardware devices of the conveyor module, i.e. motor and sensors. In this work, as illustrated Figure 2, each agent is running in a Raspberry Pi single board computer, which due to its processing, memory and wireless communication capabilities allows to support the development of decoupled and modular conveyor devices.

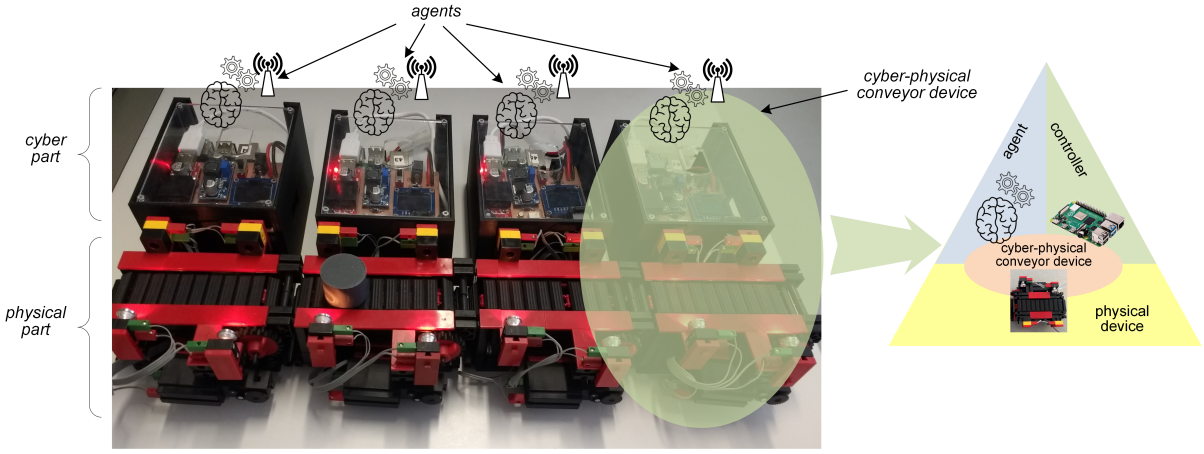

Figure 2: Cyber-physical conveyor system formed by a set of cyber-physical conveyor devices.

The logical control of a generic individual agent that is controlling one single conveyor, shown in Figure 3, was modelled by using the Petri nets formalism (Murata, 1989), which is suitable to model dynamic and concurrent processes.
Briefly, the logical control model of the conveyor agent starts by performing the setup procedure (represented by the timed transition t_2), involving the registration of the agent's skills in the

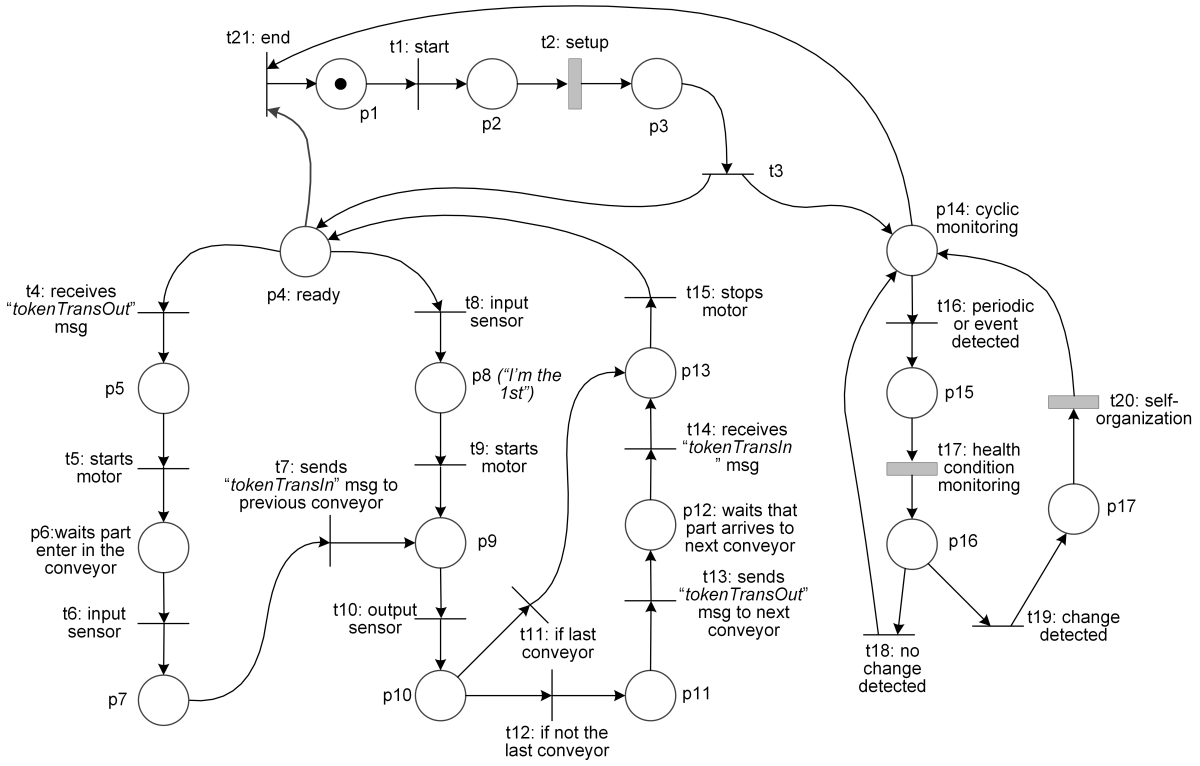

Figure 3: Generic logic control model for an individual conveyor agent.

Directory Facilitator (DF) service, the search for other conveyor agents, and the initialization of the Raspberry Pi's GPIO (General Purpose Input/Output) ports. After performing this initialization, the agent starts the execution of several behaviours running asynchronously in parallel, related to control the internal conveyor belt, monitoring the health state of the conveyor operation (transition t_{17}) and managing the self-organization process (transition t_{20}). The control of the conveyor module is achieved by determining the motor operation condition, i.e. ON or OFF, according to the input signals of the physical conveyor (that are being continuously acquired from the light sensors), and the received messages from precedent and posterior agents that ensures the synchronization according to the precedence of the sequence. As example, after receiving the *tokenTransOut* message from the previous conveyor (i.e. transition t_4), the conveyor agent starts its motor (i.e. transition t_5).

Taking advantage of the powerful mathematical foundation associated to the Petri nets formalism, the designed agent model was formally validated using a qualitative and quantitative analysis. This procedure, executed during the design phase, allowed to verify that the model is free of errors and ready to be implemented. In this work, the agent-based system was implemented by using the JADE framework (Bellifemine, Caire and Greenwood, 2007), which is a FIPA (Foundation for Intelligent Physical Agents) compliant framework that simplifies the development of agent-based solutions using the Java language. As the system comprises several conveyors, and all have the same behaviour, the designed generic agent logical control model is instantiated as many agents as the number of conveyors, which simplifies the design of such large-scale CPS.

The deployment of each agent in the Raspberry Pi platform requires to upload the agent pack-

age (i.e. the .jar file containing the agent instantiation) into a system directory, after performing some preparatory work in the Raspberry Pi, such as the installation of a Java Run-time Environment (JRE), the installation of a Java API to allow the agent to access the GPIOs and/or to system information and also the definition of the environment variables (to simplify the agent's execution from any point in the system). The implementation uses the PI4J Java API (http://pi4j.com/) that bridges the Raspberry Pi kernel into Java compliant methods, allowing to control the GPIOs and to support the access to the serial communication to gather the system/network information and the creation of event listeners. The access of the agent to the hardware is configured during its initialization behaviour (transition t_2) where the required GPIOs are provisioned and configured properly. Listeners are also added to govern the agent's actions triggered by the change of state in the input and output light sensors, according to the model illustrated in Figure 3.

Due to the JADE inherent features, one main container must be initiated before the agents' execution to contain the agent platform. Even knowing that one of the Raspberry Pi boards could be selected to host this main container, a cloud-based approach was selected to host the JADE agent-based platform. This means that no central governing mechanism is deployed into the cloud, i.e. the system is governed using a decentralized self-organization mechanism without providing any system topology, size or conveyor position into the agents. This decision allows to reach a more flexible approach permitting to remove any conveyor, releasing the user from the constant need of ensuring the main-container up-time.

4.2 Emergence of the Cyber-Physical Conveyor System

The global behaviour of the agent-based conveyor system emerges from the interaction among the several individual conveyor agents, each one contributing with its local behaviour. In particular, agents need to interact each other to determine their positions in the sequence of the conveyor system and to execute the transfer of the parts.

For this purpose, several interaction patterns were designed to exchange information with the adjacent agents regarding the position of the parts. In this context, the *tokenTransIn* message is used when the conveyed part is in the input sensor and the conveyor agent must inform the previous one (if any) that it already has the part, allowing the previous conveyor to stop its motor. Similarly, the *tokenTransOut* message is used when the part is in the output sensor and the conveyor agent must inform the successive conveyor agent (if any) that it should start its motor to properly receive the part.

The agents, in addition to these messages, exchange other types of messages between them during their cooperation processes, namely the *informIAmAlive*, *thereIsASwap*, *swapTokenFound* and *goodbye*. These messages, which are used at different stages of the agent's behaviours, will be explained during the description of self-organizing mechanisms.

The agents are communicating using the FIPA Agent Communication Language (ACL) standard and messages are being exchanged using the WiFi capability of the Raspberry Pi.

4.3 Embodied AI to Support Monitoring and Security

The use of IoT and AI allows to collect the amount of data that is being generated in each conveyor system aiming to monitor its health condition. For this purpose, each agent acquires data related to the motor and input/output sensors states, the vibration of the conveyor belt, captured using an accelerometer, the battery level and the current consumed during the system operation. The collected data uses IoT technologies, namely the MQTT (Message Queuing Telemetry Transport) protocol, that uses the Transmission Control Protocol (TCP) in a publish/subscribe schema.

The collected data is analysed in real-time to support the correct behaviour of the conveyor and also to detect abnormal situations, in preference in advance. For this purpose, amongst others, alerts are generated in case of delays in part transfers, low level of battery and excessive vibration in the conveyor belt. These alerts are generated by applying process control methods, such as Nelson rules (Nelson, 1984), that use the mean value and the standard deviation to determine if a measured variable is out of control or presents a trend that shows that the variable is near to be out of control. Figure 4 illustrates the use of two rules to detect situations where the conveyor is in a out of control condition for the acceleration on the Z axis.

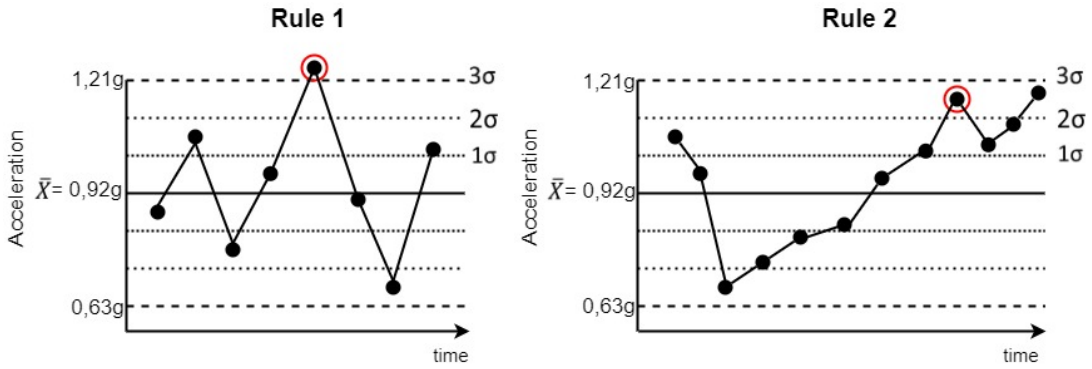

Figure 4: Rules to detect anomalies, outliers and trends, in the conveyor operation behaviour.

The Rule 1 aims to detect an outlier in the operation of the conveyor, being triggered when a value of the acceleration is greater than 3σ, with σ being the standard deviation of the acceleration (in this case the outlier will be detected if the acceleration value exceeds by 31% the average value). On the other hand, the Rule 2 allows to identify a continuous growth trend in the acceleration value, i.e. six successive increasing values in a row, allowing to detect in advance possible failures.

Besides the non-linearity, another important issue, that arises in distributed environments due to the large amount of communications, is related to the different types of security attacks that the system can be targeted, such as botnets and malware, distributed denial of service (DDoS), Man-in-the-Middle (MitM) and data breaches.

In this context, the interactions between agents to determine their position in the transportation system sequence and to transfer parts present several vulnerabilities, as the system may be attacked by an external entity or have some modified or leaked information, causing serious damage to the system. AI techniques can be incorporated into MAS to eliminate or at least to reduce the system vulnerabilities, for example using machine learning (ML) to have agents

able to use past experience to predict future decisions, which would make agents made better decisions regarding incoming message patterns.
Having this in mind, the data related to the messages exchanged between the agents was collected, namely, the sender, receiver, content, protocol, product id and timestamp. Later, the collected data is analyzed by using two different ML algorithms to detect possible attacks that may interfere in the system operation (see Figure 5).

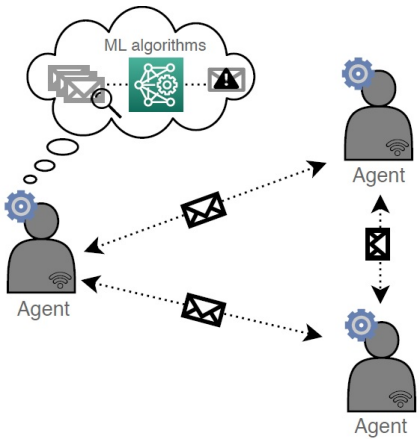

Figure 5: ML techniques incorporated in agents to detect security alerts.

The first implemented ML algorithm is the Extreme Gradient Boosting (XGBoost) classification algorithm that aims to verify if all messages exchanged during the transfer operation of one part are following the normal pattern and are not malicious messages sent by external agents that may affect system behaviour. The algorithm analyses the number of messages exchanged, average time between messages, processing time of all messages, and number of conveyors and number of protocols exchanged during part transport. The algorithm uses “gbtree” for booster and was configured with 31% of test size, the learning rate was 0.3, the number of estimators was 100 and the max_depth parameter was 8 to avoid overfitting the data training, obtaining 98.2% of accuracy, 99% of precision and 93% of recall.
The second approach consists in observing each message received separately, in case of a detection of an abnormal pattern. For this purpose, Decision Trees classification algorithm was used to analyse the receiver, content, protocol, conveyor id (token), time, number of conveyors and number of messages already exchanged data. The algorithm was configured as 30% of test size, the number of estimators was 100, and obtained 97.3% accuracy, 96% of precision and 93% of recall.
The two approaches are incorporated in the several agents, being the first one responsible to analyse the received package of messages to verify the working pattern for the complete transport of each part. In case a malfunction is detected, each exchanged message will be analysed by the second algorithm. Whenever a the algorithm predicts that a message is not reliable, the agent will ignore that message, protecting the system from possible attacks.

5 Enhancing Self-Organization Capabilities

In such modular conveyor transfer systems, where there is not a predefined sequence of conveyor modules, the agents need to determine, individually, its position in the sequence for a proper transfer operation. This need is also important when a change in the conveyor system occurs, e.g., addition, removal or swap of modules. The MAS technology allows to implement a modular cyber-physical conveyor system, but a step further is required to reach the complete pluggability and reconfigurability. For this purpose, a simple and efficient self-organization mechanism was embedded into the deployed agents' behaviour, relying in the transmission of tokens among the agents, similarly to the token passage mechanism in a 4x100m relay run.
The following sub-sections detail the several situations where the self-organization is applied.

5.1 Determining the Position in the Sequence during a Cold Start

When a conveyor agent is initiated and plugged into the conveyor transfer system, it is not aware of its position in the system sequence, requiring the execution of a proper procedure to determine its position. In this case, when a part reaches the input sensor, the conveyor agent will assume that it is the first of the system sequence, assigning itself the position 1 to the *position ID* parameter, and starts the transfer of the part by turning ON the conveyor belt. Later, when the conveyed part reaches the output sensor, the agent will propagate a *tokenTransOut* message with its *position ID* to the other conveyor agents, i.e. it will pass the *token*. The conveyor agent which *position ID* is equal to the *token + 1* will start its motor to receive the part. In case that a given conveyor does not know its position, i.e. does not have a *position ID*, it will also start its motor in order to detect if it is the next conveyor in the system. This procedure is repeated until the part reaches the last conveyor.

5.2 Adding a New Conveyor Module

The addition of a new conveyor may occur at different locations along the sequence, and it is automatically and decentralized sensed by using the interaction pattern illustrated in Figure 6. Briefly, after registering its skills in the DF service, the conveyor agent sends an *informIAmAlive* message to the other agents informing that it is ready to work. The other conveyor agents receiving this message are able to update the number of conveyors placed in the system, stored in the *conveyorCount* variable, and send a response to the new agent with the value of that variable, allowing that the new agent can also update locally this value. After this initial phase, the new conveyor agent is waiting the occurrence of one of the following two situations: i) the arrival of a part in the input sensor or, ii) the arrival of a *tokenTransOut* message. If the first case occurs without receiving any *tokenTransOut* message, this means that the new conveyor was placed at the beginning of the sequence, receiving a token value of 1. After starting its motor to transfer the part, the conveyor agent sends a *tokenTransOut* message to the other agents when the part reaches the output sensor. In the second case, when the *tokenTransOut* message is received, the conveyor agent starts its motor and waits until the part reaches its input sensor. If it occurs before a timeout, it means that the new conveyor is placed after the

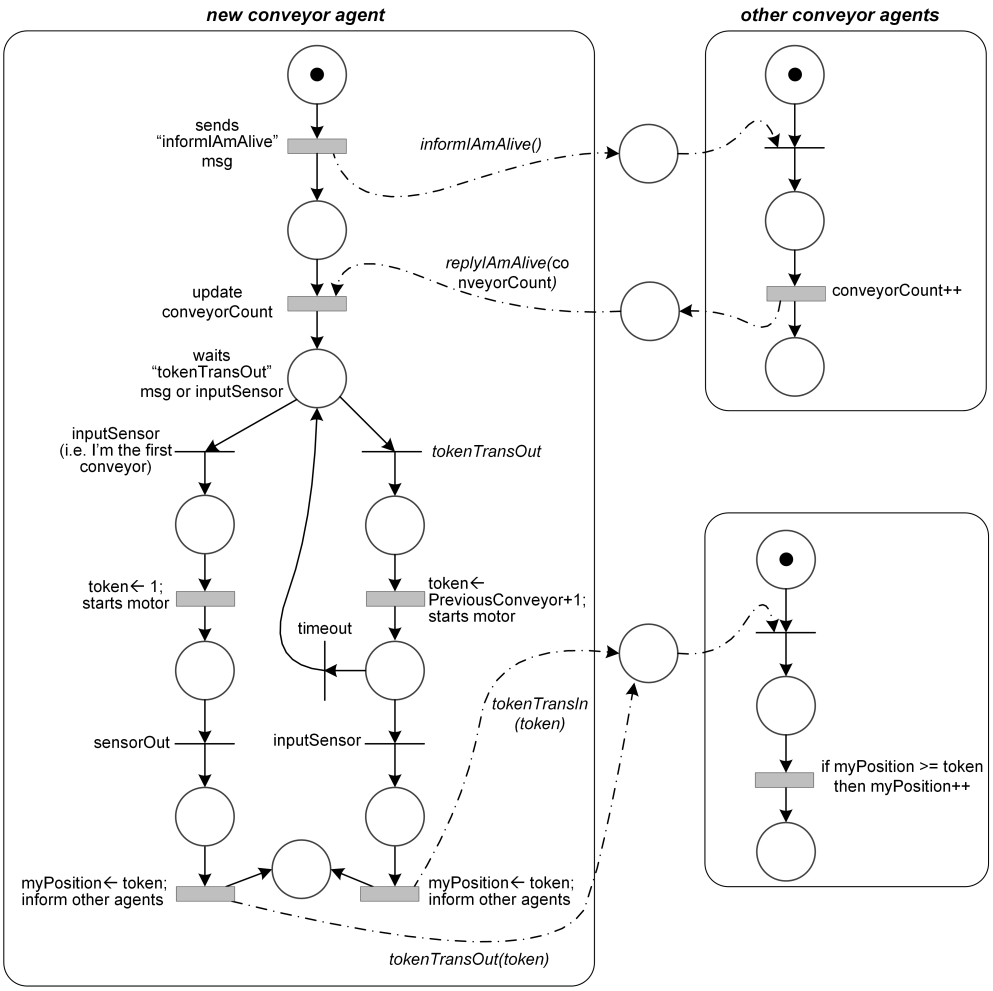

Figure 6: Self-organization mechanism for the automatic addition of a new conveyor module.

conveyor that has sent the *tokenTransOut* message. In this case, the new conveyor agent updates its location and sends a *tokenTransIn* message to the previous conveyor agent, and a *tokenTransOut* message to the other conveyor agents when the part reaches its output sensor. In this case, only the conveyor agents that are located after the introduced new conveyor will update their positions.

If the new conveyor agent detects that it is located at the end of the sequence, it will stop its motor when the part reaches the output sensor. In this case, the conveyor agents only update the individual conveyors counting without affecting the other agents.

5.3 Removing a Conveyor Module

In case of a removal of a conveyor module, the leaving conveyor agent sends a *goodbye* message to the other agents, as illustrated in Figure 7.

The conveyor agents receiving this message will decrease the *conveyorCount* variable that indicates the number of agents in the system and those placed after the removed conveyor will downward their locations by one position.

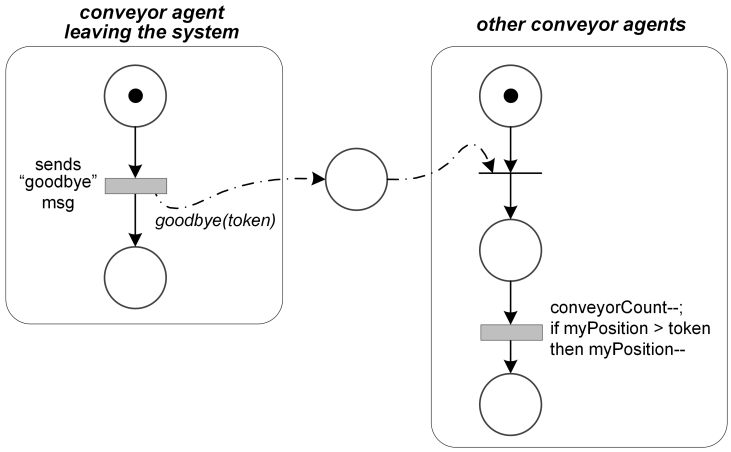

Figure 7: Self-organization mechanism for the automatic removal of a conveyor module.

5.4 Swapping Conveyor Modules

The mechanism to identify the order change of the conveyor modules requires a slightly different approach since the conveyor counting is kept and the new positions must be discovered. The self-organization mechanism to support the conveyor swap is illustrated in Figure 8. In normal operation, when a part is at the output sensor, a message is broadcasted to all the conveyor agents and is processed by the succeeding conveyor agent and discarded by the rest. When the succeeding conveyor agent detects that the part has not reached by its input sensor (by means of a timeout), it broadcasts the *thereIsASwap* message that warns for a possible order change (and containing the indication of the current position).

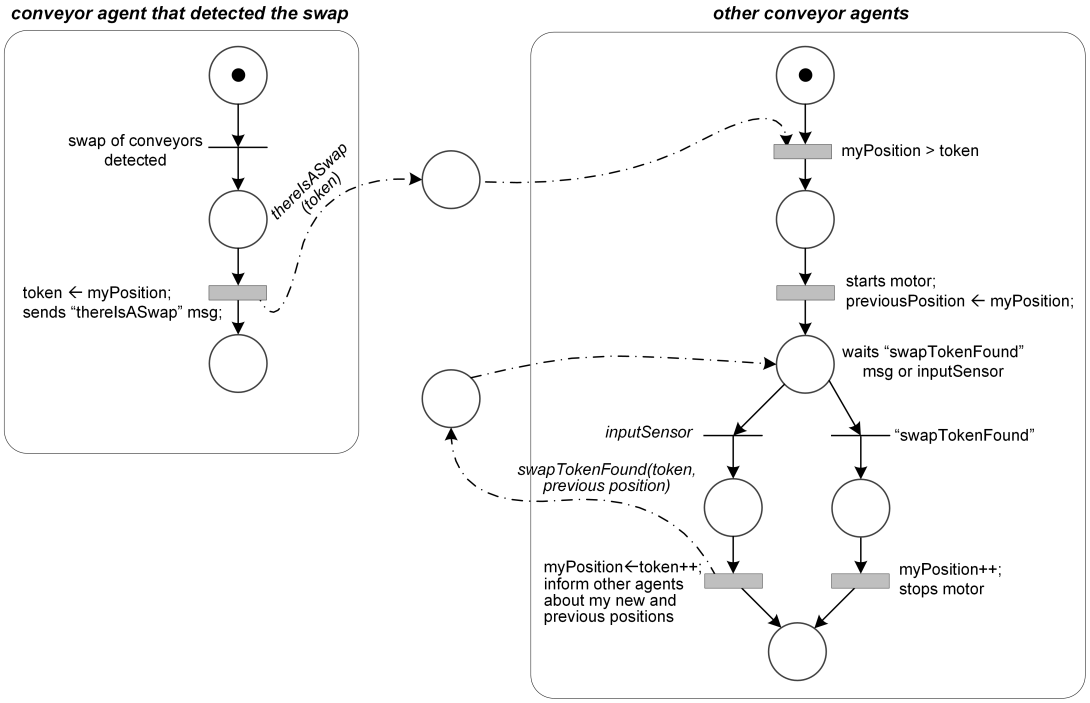

Figure 8: Self-organization mechanism for the automatic swap of conveyor modules.

All the conveyor agents that have a token higher than the received position, will start their

motors (naturally guaranteeing that they are in a valid situation, e.g., without having a part to convey). Afterwards, when a conveyor agent receives the piece at its input sensor, it will update its current system position and broadcasts a *swapTokenFound* message with its previous and new positions (updated with the position received during the *thereIsASwap* message). The other conveyor agents that receive this message only update their new positions.
This swap mechanism works perfectly for all possible conveyor agent position changes except for the case where a conveyor agent is changed to the beginning of the sequence to assume the token equal to 1. In this case, an additional rule must be implemented, which consists in detecting a conveyor agent receives the indication that a part is placed in its input sensor without being previously notified by a *TokenTransOut* message. The conveyor agent that identifies the exchange sends a *thereIsASwap* message to the other agents, and those agents that receive this message will update the token value to 0 and start the motors to re-determine the correct sequence.
This process is repeated every time a possible conveyor order has changed, governing, in a decentralized and self-organized manner, the system behaviour.

5.5 Self-organized Conveyor System Working in Practice

A video showing the system operation is located at http://youtu.be/HMG2_tTGk20. Here it is possible to observe all the aforementioned features with the capability to support the addition, removal and swap of the conveyor order on-the-fly. The presented approach shows how to use of multi-agent systems, enhanced with self-organization capabilities, to govern the conveyor system in a distributed and decentralized manner, allowing to design and deploy industrial complex large-scale systems, exhibiting pluggability and reconfigurability on-the-fly. The absence of a central control logic node also increases the system robustness and scalability by eliminating single-point of failure situations.

6 Conclusions

In the era of the 4th industrial revolution, novel modular and smart solutions that provide pluggability, robustness, flexibility, responsiveness, and reconfiguration on-the-fly are strongly demanded. Such solutions are designed to follow the CPS approach that acts as a backbone infrastructure to integrate key enabling technologies like IoT and AI. MAS is an enabler to develop industrial CPS due to its capability to distribute intelligence by a society of autonomous control nodes, better facing the dynamic adaption to the condition changes without external intervention. The agent technology is fascinating essentially because it is suitable to build complex systems with autonomous building blocks, each one incorporating the capability to take decisions without hierarchy but contributing to the best of the system. It is also important to note that there is not a centralized structure or a client-server schema but instead, the system behaviour emerges from the interaction among distributed agents. However, the truly on-the-fly reconfiguration can only be achieved if agents will embed AI and self-organization techniques, allowing the development of industrial modular and reconfigurable solutions, aligned with the Industry 4.0 principles.

This paper describes the implementation of a modular and self-organized conveyor transfer system that uses MAS, enhanced with self-organization techniques, as an enabler for its operation and reconguration. For this purpose, the system comprises several individual cyber-physical conveyor devices, each one composed by a conveyor module, constituting the physical part, and a logical part, implemented using the agent technology and deployed in a Raspberry Pi board. The agent-based system was implemented using JADE, with the agents embodying a simple and distributed self-organization mechanism, comprising several simple rules, to govern the system operation facing the condition change. The aggregation of different individual cyber-physical conveyor devices allows transferring parts from an initial position to the final position, by means of a decentralized control approach.

With this approach, the re-organization of conveyors is simplified and can be performed on-the-fly, which means that it is not necessary to stop, reprogram and restart the system in case of adaptation to the condition change, e.g., addition or removal of a conveyor, or even a change in the conveyor sequence. Additionally, the incorporation of IoT and AI technologies enables the connectivity between the digital and physical systems, contributing for the real-time data collection and monitoring. AI techniques also allow to identify possible threats by analysing the messages exchanged between the agents, protecting the system from potential external attacks.

This cyber-physical conveyor system was used in the practical learning classes of the last summer school on intelligent agents in automation, held in Bragança, Portugal, providing hands-on experience to the participants on deploying agent-based systems and developing self-organization mechanisms.

As future work, more powerful AI techniques should be embedded in the agents to support advanced monitoring, reconfiguration and security functionalities.

References

Barbosa, J., Leitão, P., Adam, E. and Trentesaux, D. 2015. Dynamic Self-organization in Holonic Multi-agent Manufacturing Systems: The ADACOR Evolution, *Computers in Industry* **66**: 99–111.

Barbosa, J., Leitão, P. and Teixeira, J. 2017. Empowering a Cyber-Physical System for a Modular Conveyor System with Self-organization, *in* T. Borangiu, D. Trentesaux, A. Thomas and O. Cardin (eds), *Service Orientation in Holonic and Multi-Agent Manufacturing, Studies in Computational Intelligence*, Vol. 762, Springer, pp. 157–170.

Bellifemine, F., Caire, G. and Greenwood, D. 2007. *Developing Multi-Agent Systems with JADE*, Wiley.

Bousbia, S. and Trentesaux, D. 2002. Self-organization in Distributed Manufacturing Control: State-of-the-art and Future Trends, *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 440–451.

Bussmann, S., Schild, K. and Baumgaertel, H. 2000. Self-Organizing Manufacturing Control: An Industrial Application of Agent Technology, pp. 87–94.

Kagermann, H., Wahlster, W. and Helbig, J. 2013. Securing the Future of German Manufacturing Industry: Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0, *Technical report*, ACATECH – German National Academy of Science and Engineering.

Lee, E. 2008. Cyber physical systems: design challenges, *11th IEEE international symposium on object/component/service-oriented real-time distributed computing*, pp. 440–451.

Leitão, P. 2009. Agent-based Distributed Manufacturing Control: A State-of-the-art Survey, *Engineering Applications of Artificial Intelligence* **22**(7): 979–991.

Leitão, P. and Restivo, F. 2006. ADACOR: A Holonic Architecture for Agile and Adaptive Manufacturing Control, *Computers in Industry* **57**(2): 121–130.

Leitão, P. 2009. Holonic Rationale and Bio-inspiration on Design of Complex Emergent and Evolvable Systems, *in* A. Hameurlain, J. Küng and R. Wagner (eds), *Transactions on Large Scale Data and Knowledge Centered Systems, Lecture Notes in Computer Science*, Vol. 5740, Springer, pp. 243–266.

Leitão, P., Karnouskos, S. and Colombo, A. 2016. Industrial Automation based on Cyber-Physical Systems Technologies: Prototype Implementations and Challenges, *Computers in Industry* **81**: 11–25.

Miller, P. 2007. The genius of swarms, *National Geographic Magazine* .

Murata, T. 1989. Petri nets: Properties, Analysis and Applications, *Proc. IEEE* **77**(4): 541 – 580.

Nelson, L. S. 1984. The Shewhart Control Chart—Tests for Special Causes, *Journal of Quality Technology* **16**(4): 237–239.

Valckenaers, P., Hadeli, Kollingbaum, M., Brussel, H. and Bochmann, O. 2002. Stigmergy in Holonic Manufacturing Systems, *Journal of Integrated Computer-Aided Engineering* **9**(3): 281–289.

Vyatkin, V. 2011. IEC 61499 as Enabler of Distributed and Intelligent Automation: State-of-the-Art Review, *IEEE Transactions on Industrial Informatics* **7**: 768 – 781.

Wooldridge, M. 2002. *An Introduction to Multi-Agent Systems*, John Wiley and Sons.

Zambrano, G., Bonte, T., Prabhu, V. and Trentesaux, D. 2014. Reducing Myopic Behavior in FMS Control: A Semi-Heterarchical Simulation Optimization Approach, *Simulation Modelling Practice and Theory* **46**: 53–75.