ORIGINALARBEIT Elektrotechnik & Informationstechnik (2018) 135/3: 270–277. https://doi.org/10.1007/s00502-018-0614-7

SAMBA – an architecture for adaptive cognitive control of distributed Cyber-Physical Production Systems based on its self-awareness

L. C. Siafara, H. Kholerdi, A. Bratukhin, N. Taherinejad, A. Jantsch

Factories in Industry 4.0 are growing in complexity due to the incorporation of a large number of Cyber-Physical System (CPSs) which are logically and often physically distributed. Traditional monolithic control and monitoring structures are not able to address the increasing requirements regarding flexibility, operational time, and efficiency as well as resilience. Self-Aware health Monitoring and Bio-inspired coordination for distributed Automation systems (SAMBA) is a cognitive application architecture which processes information from the factory floor and interacts with the Manufacturing Execution System (MES) to enable automated control and supervision of decentralized CPSs. The proposed architecture increases the ability of the system to ensure the quality of the process by intelligently adapting to rapidly changing environments and conditions.

Keywords: self-awareness; cognitive systems; dynamic clustering; autonomous collaborating objects; system health monitoring

SAMBA – eine Architektur zur adaptiven kognitiven Kontrolle verteilter cyber-physischer Produktionssysteme basierend auf Self-Awareness.

Industrie 4.0-Fabriken nehmen rasch an Komplexität zu aufgrund der Einbeziehung einer großen Anzahl von cyber-physischer Systeme (CPS), die logisch und oft physisch verteilt sind. Traditionelle monolithische Kontrolle und Überwachungsstrukturen sind nicht in der Lage, den steigenden Anforderungen hinsichtlich Flexibilität, Betriebszeit und Effizienz sowie auch Belastbarkeit gerecht zu werden. „Self-Aware Health Monitoring and Bio-inspired coordination for distributed Automation systems" (SAMBA) ist eine kognitive Anwendungsarchitektur, die Informationen von der Fabrik verarbeitet und mit dem Manufacturing Execution System (MES) zur automatisierten Kontrolle und Überwachung von dezentralen CPS interagiert. Die vorgeschlagene Architektur erhöht die Fähigkeit eines Systems, durch intelligente Anpassung an eine sich schnell verändernde Umgebung bzw. Bedingungen die Qualität des Prozesses zu gewährleisten.

Schlüsselwörter: Self-Awareness; kognitive Systeme; dynamisches Clustering; Autonomous Collaborating Objects; System Health Monitoring

Received January 30, 2018, accepted April 14, 2018, published online June 4, 2018

1. Introduction

Highly efficient production requires high degrees of flexibility, adaptiveness, and responsiveness in order to achieve high quality and versatility of manufacturing processes [1]. To reduce lead time, previous approaches have focused on rigid and deterministic automated production environments, which minimize disturbances during operation [2]. However, the increasing structural complexity of production systems, due to the growing number of CPSs and distributed heterogeneous components in the loop, decrease the deterministic nature of production processes and require agile controls capable of prediction and timely reaction to disturbances [3]. To achieve flexibility while enhancing system performance, real-time information from the shop floor shall be integrated into the control system. Moreover, the optimal reaction should be decided with respect to the system goals, which may themselves change during the operation. To further assure quality, an important task is the continuous measurement of individually varying product properties in early stages [4]. In view of these requirements, researchers have proposed various methods of intelligent sensing, self-organization, and self-optimization and a number of sophisticated cognitive architectures [5, 6].

Even though the developments of technology have improved the robustness and resiliency of Cyber-Physical Production System (CPPSs) in comparison to conventional automation systems, new expectations such as self-diagnosis and prognosis, self-repair, self-discovery and self-configuration, predictability as well as safety [7] are rising. We categorize these expectations into three different challenges. (i) The first challenge is the self-aware health monitoring which means self-observation and fault diagnosis. (ii) The second challenge regards the complication of the decision-making process

Siafara, Lydia Chaido, TU Wien, Institute of Computer Technology, Gußhausstraße 27–29, 1040 Vienna, Austria; **Kholerdi, Hedyeh,** TU Wien, Institute of Computer Technology, Gußhausstraße 27–29, 1040 Vienna, Austria; **Bratukhin, Aleksey,** Danube University Krems, Center for Integrated Sensor Systems, Viktor Kaplan Straße 2 E, 2700 Wiener Neustadt, Austria; **Taherinejad, Nima,** TU Wien, Institute of Computer Technology, Gußhausstraße 27–29, 1040 Vienna, Austria; **Jantsch, Axel,** TU Wien, Institute of Computer Technology, Gußhausstraße 27–29, 1040 Vienna, Austria (E-mail: axel.jantsch@tuwien.ac.at)

required for autonomy, robustness, and safety in the system. (iii) Finally, the third challenge regards the communication and decentralized existence of information in a distributed system, operating in known or unknown environments.

In this paper, we propose an architecture designed to address these individual challenges in an integrated and comprehensive manner in order to meet the expectations of modern production systems. Therefore, the proposed system aims at the dynamic observation of the environment, data abstraction, and distributed negotiation. They serve each unit of the system in detecting the faults and anomalies independently but collaboratively. Each unit uses self-configuration to achieve dynamic clustering of the system for the purpose of communication between units. Finally, the system attempts to go beyond using the existing actions for known problems and to mitigate new anomalies and problems to guarantee the safety of the whole system.

It should be born in mind that the proposed architecture is work in progress and not yet completely implemented. We have simulated the SAMBA architecture in the nxtStudio based environment and verified its main functionality, which we present here. We have implemented a portion of Self-Aware Health Monitoring (SAHM), namely health monitoring of an injective function black box using contextual information, which we tested on an AC-motor case study [8]. Moreover, we have recently studied the utility of SAMBA features in the context of hierarchical goal management and found encouraging preliminary results for their use in autonomous anomaly mitigation, self-configuration and arbitration between conflicting goals [9]. Although we are encouraged by the results so far, the main challenge will be to demonstrate that SAMBA leads to a measurable improvement in overall reliability and operational efficiency of the production system. That is the target of an ongoing FFG funded project with TU Wien, Danube University Krems (DUK), nxtControl, and AVL (Anstalt für Verbrennungskraftmaschinen List) as partners.[1]

This paper first reviews the state of the art in the areas of self-aware monitoring, cognitive systems and dynamic clustering in Sect. 2. In Sect. 3, it explains how these methods are integrated into the SAMBA architecture and presents the modules of the solution and their interfaces. Next, in Sect. 4, it goes through an exemplary scenario with disturbances to elaborate how the system handles the problem and presents a high-level simulation of the system using IEC 61499 function blocks. Finally, it discusses potential benefits of the proposed architecture and draws conclusions.

2. State of the art

2.1 Self-aware monitoring

Recent attempts to improve the efficiency, collaboration and resilience of automated systems in industry highlight the importance of the self-awareness in such systems. Self-awareness enables a system to monitor itself and its own environment to assess its situation better and make more appropriate decisions. An architecture for cyber-physical manufacturing system based on Industry 4.0 has been proposed by Lee et al. [10]. The authors studied a unified 5-level architecture. The first level deals with the data acquisition and then the self-awareness is used in the second level to monitor the health degradation of the system. In another study, self-representation for monitoring automation systems using a multi-agent structure was studied by Kaindl et al. [11]. An agent is defined as a combination of hardware and software components which represents itself and its relations to its environment in an explicit symbolic manner. The proposed system is used for self-configuration as well as monitoring and failure detection. However, no cause detection was considered in that work.

Self-monitoring has been implemented also as a hierarchical agent in a Systems-on-Chip (SoC) [12]. The proposed structure facilitates the process of monitoring parallel many-core SoCs. Cyber-Physical System-on-Chip is another platform where self-awareness has been explored [13]. There, the authors described self-awareness as the ability of a system to monitor its own internal and external behavior in order to make appropriate decisions. The superiority of the hereby proposed architecture of self-aware health monitoring over the state of the art is its ability to interpret the reliability of the data, measure the confidence of its processes, use attention for more efficient resource utilization, and perform predictions to provide more information for the decision-making process.

2.2 Cognitive systems

Flexible and reliable operation in many automated processes is achieved by the inclusion of the human operator in the loop [14]. Cognitive abilities enable humans to solve problems under uncertainty and despite the changes in the environment and tasks. Although such capabilities are common in humans, they are rarely found in industrial systems. The goal of cognitive architectures is to define a framework for designing systems with human-like intelligence; they provide a structure which enables a system to develop over its lifetime by embedding the mechanisms of perception, reasoning, action, and learning [15]. Different cognitive architectures have been applied for realizing cognitive tasks, for example, SOAR [16], LIDA [17] and ACT-R [18].

To address limitations related to incomplete sensing of the environment and inability to individually carry-out global tasks, cognitive systems often exhibit social behavior, which entails communication and cooperation or negotiation. Such an approach is adopted in cognitive radio networks, where multiple distributed sensors collaborate selectively to enhance their spectrum sensing performance and the utilization of the radio frequency spectrum [19]. Distributed cognitive systems have been studied also for the control of robots [20], and building environmental control [21]. In industrial applications, the cognitive production systems refer to highly interconnected devices with improved sensing, reasoning, learning and planning capabilities, which use knowledge-based and learning models to assess and expand their capabilities [1]. The cognitive system design paradigm provides a promising approach towards the dynamic adaptation of the processes and the continuous optimization through learning by observation of the environment and reasoning for mitigating errors and finding improved operation strategies.

A more recent cognitive architecture is the Simulation of Mental Apparatus (SiMA) [22], which has previously been applied for cognitive control in building automation [23]. SiMA focuses on functions that generate human behavior and implements the underlying mechanisms (e.g., drives, emotions) which drive a system to exhibit a certain behavior. This behavior is thereby defined by the internal state of the system, which on its turn depends on the feedback returned by the environment on the system's actions, rather than being explicitly defined by the environmental state. Although this approach requires a higher engineering effort in the beginning due to the need for implementing the underlying driving behavioral mechanism, it can achieve a lower system complexity since the states under which a behavior is exposed do not need to be defined explicitly [24]. The SiMA theory is used as basis for the design of the

[1] SAVE *(Self-monitoring-based process Adaptation for quality assurance in heterogeneous VErsatile manufacturing)*, funded by FFG under contract 864883.

cognitive module in SAMBA, which is discussed in detail in the next section.

2.3 Dynamic clustering

A key to an efficient performance in distributed systems is communication among system components. The concept of clustering was devised in conjunction with the need to adapt to high complexity and the dynamic nature of emerging manufacturing control systems. It originated from the Holonic Manufacturing Systems (HMS) concept [25] and the MetaMorph approach of HMS [26], based on the idea of expanding problem-solving clusters. Each cluster is a dynamically created community of intelligent system components that cooperate with each other to obtain enough information to solve a problem. Each member of a cluster has a set of algorithms which enable recognition of a problem and finding a solution. A flexible structure is one of the advantages of a distributed clustering concept that allows solving complex problems in a scalable manner. Clustering has found its use in a variety of industrial applications such as mitigating the complexity of production orders [27], knowledge propagation of thermal modeling [28], ad-hoc wireless networks [29] for optimizing the routing protocol [30], and in other concepts based on swarm intelligence [31, 32].

The proposed architecture uses the concept of distributed clustering for establishing relevant connections between independently acting entities in the system. Similar to [27], it establishes connections between the system components according to the production order structure. In this case, however, the goal (rather than decision-making) is more relevant to knowledge propagation [28] and therefore, the architecture presents a novelty over the current state of the art.

3. Proposed architecture

Figure 1 illustrates the architecture of SAMBA and its major modules. The logical unit of an entity is an Autonomous Cooperating Object (ACO). The ACO learns locally the specifics of the environment and the actions it can take. Further, it exhibits social behavior, since it interacts with other ACOs located in the same environment. To decide on its agenda, it takes into consideration its own goals, the environmental context and the requests from other ACOs. The global behavior emerges out of the interactions among the ACOs.

3.1 System architecture

Each ACO consists of three SAMBA-specific functional units in addition to the operation module: *Operation Module* serves as the interface to the hardware. It connects to the plant controller and forms a major part of the by providing symbolized sensor data and receiving actuator ACO commands. Often, this unit sits on the legacy controller. *SAHM* monitors the system operation and in case of anomalies, it proposes possible causes. To this end, SAHM performs fault diagnosis and data abstraction. As each ACO is observing only its local environment, other ACOs can request or may be requested to provide further information through the external manager. *Internal Manager* is the decision-making entity of the system. It receives health status information from SAHM and external requests from other ACOs through the external manager. At a certain state, there might be conflicting action requests and the internal manager shall decide which goal is prioritized. *External Manager* provides an implicit connectivity among the relevant ACOs via distributed clustering based on the production order structure and the physical layout of the shop floor. It facilitates transparent negotiations between one ACO and other ACOs in the cluster. In the rest of this section, we discuss the main functionalities of each module.

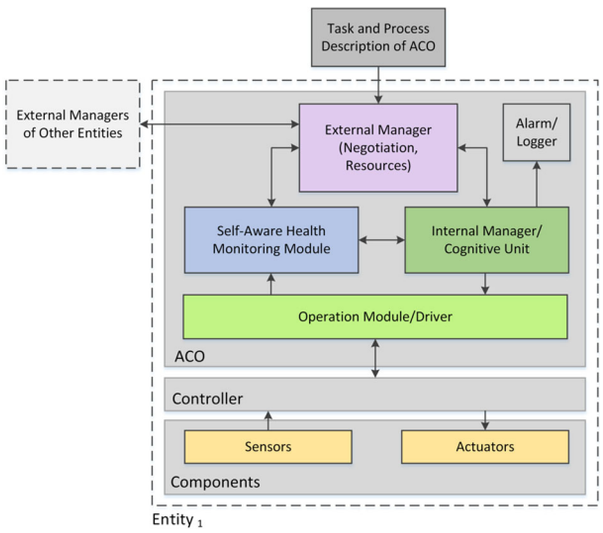

Fig. 1. The architecture of SAMBA depicting the structure of one Entity

3.2 Self-aware knowledge extraction and representation

SAHM aims at parallel data *abstraction* and *health status analysis* (anomaly detection) to provide the system with a suitable observation required for self-awareness [33]. The data is abstracted in a normal situation when the sensors' functionality is correct. However, when some of the sensors are faulty, the intention is to request the equivalent data from adjacent entities. The *health status analysis* approach mostly relies on the self-observation, however, sometimes negotiation with other entities increases the knowledge of an entity regarding an anomaly. Therefore, the proposed self-aware system monitors various health parameters in a distributed manner. These parameters are used to classify the health status of the entity into a predefined state. The input parameters are the data of the sensors that have been abstracted. These sensors may belong to an entity itself or to other entities.

The goal of *abstraction* is to reduce the size of data as well as to extract knowledge which is beneficial for the further processes in SAHM, the internal manager, or other entities. It deals with the sensor data and also other specifications about the hardware part of the entity, including the nominal range and the reliability of each component. Abstraction has several functional states, e.g., abstraction of sensor data, providing a response to the internal manager's request for specific data as well as providing specific data to the external manager, or processing the data provided from another external manager.

The goal of *health status analysis* is to detect the anomaly (as well as its characteristics, causes, and effects), and to inform the internal manager. It also deals with the requests from the external manager regarding the health and operational status of the entity. *Health status analysis* includes anomaly detection, anomaly specification, cause diagnosis and prediction blocks. The abstracted data first enters the anomaly detection. The type of anomaly and the value of the reliability are the outputs which are forwarded to the other three blocks. The detected anomaly triggers the anomaly specification block which estimates the features of this anomaly, such as the rate of wear-out deterioration. Cause diagnosis detects the reason(s) of the anomaly. This block requires the abstracted data, type of anomaly and features of the anomaly (if they exist) as well as complementary information from other ACOs. If the *health status analysis* suspects the existence of an external cause, a data request is

sent to the external manager, which can ask for the data and health status of the other involved ACOs. Another output of this block is the new value of specification for the reliability of the components which is transferred to the *abstraction* block. Finally, the prediction block receives the abstracted data, type of anomaly, its features, and cause(s), and predicts the future effect of the anomaly on the entity or the system.

The frequently updated reliability information of the data and components provided from MES as well as the confidence value at the end of each step are all interpreted as metrics to measure the level of uncertainty. Different approaches are used depending on where the uncertainty is flagged. For example, if the *abstraction* unit faces unreliable data, a negotiation with the neighboring ACO(s) starts to request data from them. A self-aware Multiple Classifier System (MCS) based on the rankings of each individual classifier similar to [34] can be used to improve the performance of anomaly detection and cause diagnosis through analysis of a collecting of results and thereby choosing the optimal one to reduce the uncertainty. Cause detection can be built using a MCS whose inputs are the type of anomaly, meta-data and anomaly parameters. Multiple learning algorithms interpret the inputs and provide the result with a value of confidence. If an external cause is probable to obtain relevant external data, negotiation with other ACOs is initialized. The new external information is fed as an input to the system and a new analysis starts.

3.3 Cognitive decision-making

To select the most suitable actions with regard to the goals of an ACO, decision-making takes place on the basis of data inputs from the SAHM and the external manager. SAHM provides information regarding the current state of the ACO and its environment, whereas the external manager provides information regarding the state of other ACOs and their environment. Two input types are defined: *drives* and *percepts*. Drives are the (initial) motivations of the system and therefore, the source of the system goals. Their intensity represents the deviation from a desired goal state: the higher a drive intensity, the more urgent the goal represented by this drive. Symbols from the environment constitute the percepts of the ACO. Once percepts are generated for the current state, similar states from the memory are activated. Emotions are the internal evaluation mechanism of the system and are generated based on the current state of drives and the stored activated memories from the past. All these processes form part of the *primary process* of the system, which is characterized by the lack of reasoning functions. The primary process regulates the reactive behavior of the system and proposes actions that are able to solve urgent problems in a reactive manner [35].

In the *secondary process*, social rules which reward or penalize a behavior are tested on the current state; they represent user preferences and guide the policies that the system should comply with them. The rewards, which represent the external evaluations, along with the emotions, which are the internal evaluations, enrich the current perceived state. The enriched perceived state and the drives, which indicate goal priorities, are the inputs to the goal selection process, where it is decided which goal the system shall pursue. Next, sequences of states (episodes) similar to the current episode are activated using case-based reasoning. From the activated episodes, sequences of actions (policies) that managed to reduce the drive intensities in the past are then picked as potential options. The policy with the best performance is used to select the current action to execute. The new episode is saved in the memory by the *learning process*, which is responsible for the tasks of updating the memory with new episodes while it removes episodes that have not been activated for a long time. Decisions of the Internal Manager (IM) are evaluated and their effects are updated over time in order to account for the changing dynamics of the environment, therefore, maintaining an implicit model of the environment.

3.4 Adaptive collaboration

To compensate for the lack of a global overview due to the distributed nature of the proposed system, the concept of dynamic clustering is applied. Dynamic clustering introduces a flexible way to integrate global objectives while allowing encapsulation of core functionality of the ACO, hence, providing transparency for the decision-making. The challenge is to form clusters regarding the functional requirements of the shop floor and manufacturing instructions. Currently, there is no methodology to formalize production order ontology in relation to the anomalies. Instead of predefined rules of the existing solutions, the External Manager (EM) defines dynamic methods using learning algorithms (e.g., neural networks, and reinforcement learning) to automatically discover the cluster formation rules based on the production order, the physical layout of the shop floor, and network communication structure. A set of generic classifiers which operate on non-application-specific characteristics are defined and used by the EM to determine essential connections in the cluster and establish weighted links between ACOs based on the prioritized set of control parameters. These weighted connections are updated at runtime using backpropagation according to feedback from the evaluation of the communication and the execution of decisions.

As a result, the system achieves implicit connectivity among the relevant ACOs and facilitates transparent negotiations between SAHM and the internal manager of different ACOs in a cluster. The External Manager bases its technique on a non-deterministic relevance of connection between individual ACOs regarding a particular event. The outputs of the External Manager are used to check possible implications of local decisions on other ACOs in the community. By adjusting the relevance of the connections over time, such dependencies are dynamically adapted based on the consequences of actions.

3.5 System dynamics and interface

Communication between the units of an ACO and also between different ACOs of a cluster is necessary to achieve the functionality of the larger system described in Fig. 2. This figure highlights the wiring and content of data exchanges between units in Fig. 1. The environment consists of the yellow blocks on the top, exchanging information with the Operation Module in the ACO, and other ACOs shown at the bottom. Types of information transferred between units inside the ACO are also specified and depicted in Fig. 2. Finally, the ACO is connected to the External Manager of other ACOs, which form the other part of the environment. Note that this figure illustrates the minimum possible connections in the proposed system. Each request and its reply are labeled by an ID to simplify the process of handling them. To focus on the main system functionality, it is assumed that no communication errors occur.

The connection between the Internal Manager and the Operation Module is established for transferring the estimated set-points after the decision-making process. There is only a one-directional data flow from the Operation Module to SAHM to push all data from sensors to SAHM.

Several communication messages are separately sent from SAHM to the Internal Manager including a value representing the health status of the ACO, information on anomalies and their cause, extra data in response to Internal Manager's requests, and the action

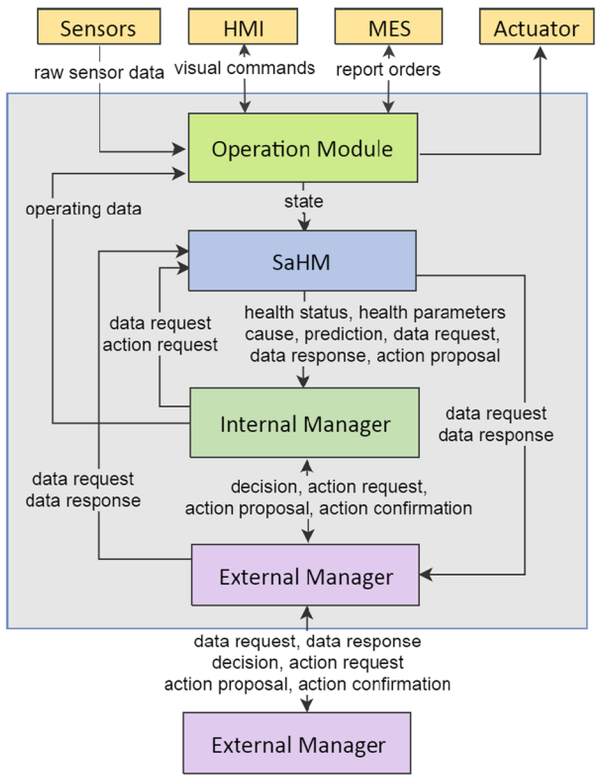

Fig. 2. Communication flow within the ACO and with other ACOs of the cluster

proposal. The action proposal responds to a request made by the Internal Manager and indicates the operational range within which the ACO can adapt its operational settings. Data and action requests are sent from the Internal Manager to SAHM. Action requests are sent to SAHM upon receipt of an action request from the external manager, in order to ask for the nominal operational range of the ACO.

The Internal Manager establishes a connection to the External Manager when the decision it makes impacts the operational parameters of another ACO. An action request is also sent when the Internal Manager decides to change an operation parameter and the negotiation needs to be initiated. Moreover, in response to an established negotiation from other ACOs, the Internal Manager sends a confirmation message to the requesting ACO in case of an agreement.

The messages from SAHM to the External Manager include data requests when SAHM detects an anomaly and needs to contact another ACO for additional information. Similar messages are exchanged in the opposite direction too. The communication mechanisms between External Managers of different ACOs are similar to the ones between the External Manager and the Internal Manager. In addition, External Managers exchange data requests to establish clusters.

4. Exemplary use case

4.1 Definition/description

To explain how the system behavior emerges out of the interactions among the individual components of the ACOs as well as through communication with other ACOs, consider an exemplary scenario with detection of a missing product. Figure 3 illustrates a potential configuration of the system; an assembly line consisting of four conveyor sections supplying two robotic arms with objects. includes three different types of entities, i.e., the conveyor sections, which transport the products, the product sorters, which sort the products by type for further processing, and the robotic arms, which process the products. Each entity consists of the hardware part (i.e., motor, sensors, and other equipment), and the logic unit (the ACO), which monitors and, if necessary, adjusts its operation. The system monitors the existence of all items in its environment. If an ACO is expecting to receive an item at a specific time but it does not, the system shall try to find the root cause of the event. To this end, it uses the observations from entities responsible for previous (or next) tasks in the process and analyzes the available information to find out if the item is missing or the sensor is not working. Afterwards, the system needs to decide whether it can adjust the settings and resolve the problem, or the operator needs to be notified.

The following tasks take place:

1. The SAHM detects the event and tries to diagnose the problem. That is, to find out whether the problem is caused by a faulty sensor or a missing product. It informs the internal manager about the event.
2. The external manager, upon notification from the internal manager, asks for sensor information from the previous/next ACOs in the process.
3. The SAHM receives the information from the other related ACOs, and analyzes all information to find out if the item is missing or the sensor is not working. This action will continue in a sequence until the source of the problem (product missing/stuck or sensor failure) is found.
4. If the problem is due to erroneous sensor data, the SAHM notifies the internal manager for this event to take the necessary actions to handle the problem.
5. If the problem is that the product is missing, due to unsynchronized speed settings of the source entities, the SAHM notifies the internal manager and the latter takes the necessary actions to handle the problem.
6. If the problem is that the product is lost, then the internal manager informs the human operator about the fault and asks for intervention.

4.2 Feasibility verification of communication architecture

To verify the feasibility of the communication architecture, we ran a simulation experiment which we present in the rest of this section. The simulation was performed in the nxtStudio runtime environment which uses the IEC 61499 standard. The standard is based on the event-driven concept of function blocks. Function blocks are independently acting components that encapsulate local functionality and communicate with each other via events and associated data. Each block implements a set of algorithms and triggers events upon completion. The main benefit of the IEC 61499 model lies in its flexibility to integrate complex systems and adapt them even at runtime.

The main goal of the simulation was to establish a communication infrastructure for testing the synergy of the individual SAMBA components. For each node in the production line of the use case (Fig. 3), a corresponding ACO was created. The general structure of an ACO is shown in Fig. 4; the ACO is represented as a composite function block with SAHM, IM, EM and OM as basic function blocks. Event inputs and outputs with associated data were defined and the communication algorithms were implemented. The resulting simulation provides a flexible environment for testing the underlying algorithms of SAMBA that can be used as an application independent runtime

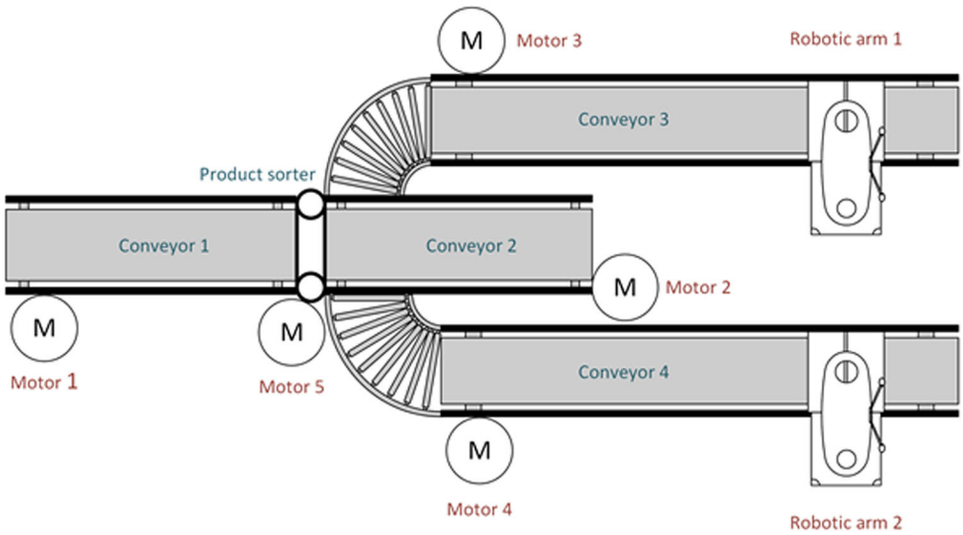

Fig. 3. Shop floor configuration in the missing product use case

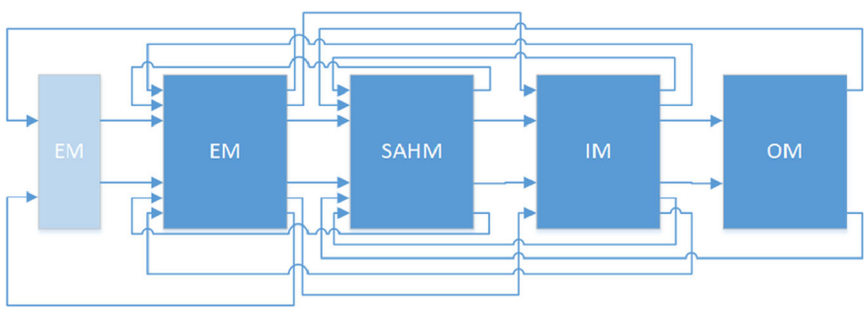

Fig. 4. ACO function-block structure

environment. Due to the modular nature of the IEC 6149 standard, a variety of use cases and algorithms can be integrated and tested to analyze the optimal configuration of algorithms and parameters for a particular application. The simulation results showed the satisfying level of scalability and flexibility of the proposed architecture and its ability to integrate required algorithms for data acquisition and decision-making.

5. Discussion

SAMBA is an architecture for CPPS that monitors the production process and reacts to deviations, either through automatic compensations or by informing the operator. It is designed to operate as a middleware on the top of existing (legacy) systems adding a layer of intelligence to the CPPS. Intelligence in this specific context is defined as the ability of the system to be context-aware and self-aware, to be proactive and social, acting within a certain degree of autonomy in a changing environment. To this end, SAMBA builds upon the concept of the ACO, as presented above. The behavior of the large system emerges out of the cooperation of individual ACOs, which dynamically form clusters and collaborate on demand while pursuing their own goals as well as the goals of the global system. To compensate for the lack of central supervision, on one hand, ACOs communicate with each other to enhance their knowledge and understand the context of their observations. On the other hand, they negotiate to align their actions in order to efficiently achieve the global system goals.

The design of SAMBA addresses following challenges which modern industrial production process faces:

1. *Increased adaptivity and reduced engineering effort:* to interpret the context of operation with minimum a priori knowledge available, semantic enhancement of the available information is used. This is enabled by self- and context-awareness concepts such as confidence, attention, data-reliability, and history, combined with intelligent data analysis, such as data abstraction and scattered-data fusion.
2. *Quality assurance and resilience enhancement through fault diagnosis and prognosis:* in a distributed system no single (sub)system knows the complete state of the overall system. Communication with other ACOs takes place to enhance local information, and to increase confidence about local knowledge. This also helps the system to obtain awareness of the bigger context in order to detect, analyze, predict, and mitigate errors, faults, and failures.
3. *Enhanced autonomy and intelligence for mitigation of anomalies in operation:* in complex manufacturing systems (such as CPPSs) thorough and precise modeling of the system and its environment is challenging. Cognitive systems are adept at making decisions efficiently despite the lack of a complete or precise model. Cognitive decision-making improves the efficiency of the system in using extracted knowledge about each ACO, its neighbors in the cluster, the overall system, and possible courses of action, to

re- or pro-actively change the CPPS, regarding the timing constraints at hand. Especially by mitigation of failures, degraded health, or anomalies, it ensures the quality of the product and process in adaptive heterogeneous manufacturing systems.

4. *Dynamic clustering based on environment discovery:* dynamic cluster building concepts which take into account the production orders parsing allow effective negotiation and propagation of mitigation measures. They also enable representation of the overall production environment beyond its locally accessible information.

SAMBA aims to introduce a new design paradigm in the industrial environment which, thanks to the generic nature of the proposed architecture, allows integration of legacy systems in manufacturing with considerably less effort compared to the state of the art. The generic nature of the architecture makes it possible to apply it independent of the specifics of the entities it controls. It is expected to reduce the downtime and frequency of maintenance interventions by health monitoring, predictive analysis and mitigation of behavioral deviations. Next steps in the development of SAMBA include the detailed design of the algorithms for the modules of the architecture and testing of the emergent system performance. In the planned follow-up project, *SAVE*, the challenge will be to demonstrate – quantitatively – that SAMBA improves the overall system performance in terms of reliability and operation efficiency.

6. Conclusion

In this paper, we presented SAMBA, an architecture for versatile heterogeneous CPPS with focus on predictive analysis, autonomous health monitoring, and error mitigation in industrial manufacturing processes. The synergies provided by these functionalities reduce system reaction time and engineering efforts through adjustment of production steps to compensate for the deviations. SAMBA increases the ability of the system to assure the quality of the product and the process by intelligently adapting to rapidly changing environments. The architecture is mainly developed for distributed CPPS, however, its generic and modular design enables its future application to different industrial systems with minimum effort for adaptation.

Acknowledgements

Open access funding provided by TU Wien (TUW). We gratefully acknowledge the financial support provided to us by the BMVIT/FFG under the program *ICT of the Future* in the project SAMBA (contract number: 855426).

References

1. Zaeh, M. F., Reinhart, G., Ostgathe, M., Geiger, F., Lau, C. (2010): A holistic approach for the cognitive control of production systems. Adv. Eng. Inform., 24(3), 300–307.
2. Shea, K. (2010): The cognitive factory.
3. European Factories of the Future Research Ass. et al. (2013): Factories of the future: Multi-annual roadmap for the contractual ppp under horizon 2020. Pub. office of the Eur. Union: Brussels.
4. Armengaud, E., Sams, C., von Falck, G., List, G., Kreiner, C., Riel, A. (2017): Industry 4.0 as digitalization over the entire product lifecycle: opportunities in the automotive domain. In Eur. conf. on software process improvement (pp. 334–351). Berlin: Springer.
5. Zhang, Y., Qian, C., Lv, J., Liu, Y. (2017): Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor. IEEE Trans. Ind. Inform., 13(2), 737–747.
6. Park, H.-S., Tran, N.-H. (2012): An autonomous manufacturing system based on swarm of cognitive agents. J. Manuf. Syst., 31(3), 337–348.
7. Monostori, L. (2014): Cyber-physical production systems: roots, expectations and r&d challenges. Proc. CIRP, 17, 9–13.
8. Götzinger, M., TaheriNejad, N., Kholerdi, H. A., Jantsch, A. (2017): On the design of context-aware health monitoring without a priori knowledge; an ac-motor case-study. In 2017 IEEE 30th Canadian conf. on electrical and comp. eng. (CCECE) (pp. 1–5).
9. Jantsch, A., Anzanpour, A., Kolerdi, H., Azimi, I., Siafara, L. C., Rahmani, A. M., Taherinejad, N., Liljeberg, P., Dutt, N. (2018): Hierarchical dynamic goal management for IoT systems. In Proc. of the IEEE int. symp. on quality electronic design, USA.
10. Lee, J., Bagheri, B., Kao, H.-A. (2015): A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufac. Lett., 3, 18–23.
11. Kaindl, H., Vallée, M., Arnautovic, E. (2013): Self-representation for self-configuration and monitoring in agent-based flexible automation systems. IEEE Trans. Syst. Man Cybern., 43(1), 164–175.
12. Guang, L., Plosila, J., Isoaho, J., Tenhunen, H. (2010): Hierarchical agent monitored parallel on-chip system: a novel design paradigm and its formal specification. Int. J. Embed. Real-Time Commun. Syst., 1(2), 85–105.
13. Sarma, S., Dutt, N., Gupta, P., Nicolau, A., Venkatasubramanian, N. (2014): On-chip self-awareness using cyberphysical-systems-on-chip (CPSOC). In Proc. intl. conf. on hardware/software codesign and system synthesis (p. 22). New York: ACM.
14. Zhao, Y. F., Xu, X. (2010): Enabling cognitive manufacturing through automated on-machine measurement planning and feedback. Adv. Eng. Inform., 24(3), 269–284.
15. Vernon, D. (2017): Two ways (not) to design a cognitive architecture. Cognitive Robot Architectures, 42.
16. Laird, J. E. (2012): The Soar cognitive architecture. Cambridge: MIT Press.
17. Snaider, J., McCall, R., Franklin, S. (2011): The lida framework as a general tool for agi. In Int. conf. on AGI. Berlin: Springer.
18. Taatgen, N. A., Lebiere, C., Anderson, J. R. (2006): Modeling paradigms in act-r. Cognition and multi-agent interaction: from cognitive modeling to social simulation, pp. 29–52.
19. Yucek, T., Arslan, H. (2009): A survey of spectrum sensing algorithms for cognitive radio applications. IEEE Commun. Surv. Tutor., 11(1), 116–130.
20. Kawamura, K., Peters, R. A. II., Bodenheimer, R. E., Sarkar, N., Park, J., Clifton, C. A., Spratley, A. W., Hambuchen, K. A. (2004): A parallel distributed cognitive control system for a humanoid robot. Int. J. Humanoid Robot., 1(01), 65–93.
21. Kollmann, S., Siafara, L. C., Schaat, S., Wendt, A. (2016): Towards a cognitive multi-agent system for building control. Proc. Comput. Sci., 88, 191–197.
22. Schaat, S., Wendt, A., Kollmann, S., Gelbard, F., Jakubec, M. (2015): Interdisciplinary development and evaluation of cognitive architectures exemplified with the SIMA approach. In EAPCogSci.
23. Zucker, G., Habib, U., Blöchle, M., Wendt, A., Schaat, S., Siafara, L. C. (2015): Building energy management and data analytics. In EDST, 2015 int. symposium (pp. 462–467).
24. Wendt, A., Kollmann, S., Siafara, L., Biletskiy, Y. (2018): Usage of cognitive architectures in the development of industrial applications. In ICAART.
25. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P. (1998): Reference architecture for holonic manufacturing systems: Prosa. Comput. Ind., 37(3), 255–274.
26. Kühnle, H. (2009): Distributed manufacturing: paradigm, concepts, solutions and examples. Berlin: Springer.
27. Babiceanu, R. F., Chen, F. F. (2006): Development and applications of holonic manufacturing systems: a survey. J. Intell. Manuf., 17(1), 111–131.
28. Bratukhin, A., Nagy, A., Mahmood, A. (2015): Distribution of control functionality in energy-aware industrial building environment. In WFCS, IEEE world conf. (pp. 1–8). New York: IEEE Press.
29. Abbasi, A., Younis, M. (2007): A survey on clustering algorithms for wireless sensor networks. Comput. Commun., 30(14), 2826–2841.
30. Liu, X. (2012): A survey on clustering routing protocols in wireless sensor networks. Sensors, 12(8), 11113–11153.
31. Saleem, M., Di Caro, G. A., Farooq, M. (2011): Swarm intelligence based routing protocol for wireless sensor networks: survey and future directions. Inf. Sci., 181(20), 4597–4624.
32. Kennedy, J. (2006): Handbook of nature-inspired and innovative computing (pp. 187–219). Berlin: Springer.
33. Taherinejad, N., Jantsch, A., Pollreisz, D. (2016): Comprehensive observation and its role in self-awareness – an emotion recognition system example. In Conf. on comp. science and inf. sys.
34. Kholerdi, H. A., TaheriNejad, N., Jantsch, A. (2018): Enhancement of classification of small data sets using self-awareness – an iris flower case-study. In Proc. of the int. symp. on circuit and systems (ISCAS), Florence, Italy.
35. Zucker, G., Wendt, A., Siafara, L., Schaat, S. (2016): A cognitive architecture for building automation. In Ind. electronics society, IECON 2016 (pp. 6919–6924). New York: IEEE Press.

Authors

Lydia Chaido Siafara
received her degree in electrical engineering and computer technology from Aristotle University of Thessaloniki, Greece, and the master's degree in building science and technologies from Vienna University of Technology (TU Wien) in 2014. Since 2015 she has been working as a project assistant at TU Wien, where she was responsible for the KORE project, dealing with the application of human-inspired cognitive systems in building automation. Her work is concerned with the development of intelligent agents and their application in the field of automation systems. Her research interests include artificial intelligence, cognitive systems and machine learning.

Hedyeh Kholerdi
received her master's degree in the field of Image Processing from Babol University of Technology, Iran. Her master's thesis is titled as "Drowsiness detection using image processing technique inspired by the human visual system". She joined TU Wien, Institute of Computer Technology (ICT) as a university assistant for Embedded Systems and Online VHDL Programming courses. She was also a project assistant working on self-aware health monitoring in scalable distributed systems. The main focus was on research in requirement of creating a contextual awareness in a distributed autonomous system using learning approaches.

Aleksey Bratukhin
received his master's degree at Perm Technical University in Russia in 1998. From 2000 to 2006 he worked at Vienna University of Technology with the focus on software agent systems and distributed control in respect to vertical integration in the area of plant automation, where he received his doctorate degree in 2006. In that year he joined the Center for Integrated Sensor Systems. His current research focus is in the area of distributed control systems, artificial intelligence, machine learning and modeling in industrial and building automation with the focus on efficient energy use.

Nima Taherinejad
is a Ph.D. graduate of The University of British Columbia, Vancouver, BC, Canada. He is currently a Post-doctorate University Assistant at TU Wien, Vienna, Austria, where his main areas of research include systems on chip, self-awareness in cyber-physical systems, embedded systems, and robotics. He was the general chair and TPC chair of MobiHealth 2017, chair of the AUTO21 HQP Advisory Committee and a member of board of directors of AUTO21 Center of Excellence. He has authored a book and published in/served as a reviewer for various journals and conferences. Dr. Taherinejad has received several awards and scholarships from universities and conferences he has attended.

Axel Jantsch
received the Dipl.-Ing. and Dr. techn. degrees from TU Wien, Vienna, Austria, in 1988 and 1992, respectively. From 1997 to 2002, he was an Associate Professor with the KTH Royal Institute of Technology, Stockholm, Sweden, where he was also a Full Professor of Electronic Systems Design from 2002 to 2014. Since 2014, he has been a Professor with the Institute of Computer Technology, TU Wien. He has authored over 300 articles and one book in the areas of VLSI design and synthesis, HW/SW codesign and cosynthesis, networks-on-chip, and self-awareness in Cyber-Physical Systems.