ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
https://doi.org/10.17559/TV-20160117100949
Preliminary communication

# New Solution of Abstract Architecture for Control and Coordination Decentralized Systems

Branislav MICIETA, Lukas DURICA, Vladimira BINASOVA

**Abstract:** This paper contains a new approach that combines the advantages and disadvantages of suppressing hierarchical and heterarchical control architectures, creating a semi-heterarchical (holonic) control architecture. The degree of subordinate unit autonomy changes dynamically, depending on the presence of a system disruption, and its scope allows for a smooth transition from hierarchical to heterarchic control architecture in subordinate units. We have proposed a representation of the dynamic degree of autonomy and its possible application to subordinate units, which are, in our case, one-directional Automated Guided Vehicles (AGVs) and are guided by magnetic tape. In order to achieve such a semi-heterarchic management architecture with a dynamic degree of autonomy, approaches such as smart product, stymergic (indirect) communication, or basic principles of holon approach have been implemented.

**Keywords:** advanced industrial engineering; biologically inspired techniques; multi-agent system; simulation; sustainable production

## 1 INTRODUCTION

In the last few decades, new production requirements have arisen and new requirements have not been met by traditional production systems. Production systems have begun to require the ability to respond to changing customer demands and hence the need to produce customizable products at the lowest cost, with high quality, in such a way that companies can maintain their competitiveness.

These customer requirements have been transformed into the required features of the new generation Factory of Future [1] manufacturing systems, namely: real-time production system response, fault tolerance, system flexibility and scalability (adding, removing and increasing system performance) decision-making decentralization, decision-making and planning, plug & produce, flexibility, agility, or simplification of control software.

The advantage of traditional hierarchical control architectures (centralized and hierarchical architectures) is their simple organizational structure (with a small number of managed units), the ability to optimize system performance and the ability to predict future states (in the absence of abnormal system states).

However, these types of control architectures cannot guarantee the demanding performance requirements of a system that requires real, dynamic change in production and logistics with all system constraints (internal changes to the system, changes in the environment) as well as changing customer demand for customizable, price with maintaining a competitive market. The complexity and rigidity of this type of architecture increase with the number of system components, making it difficult to scalable, modify and maintain the system, or to make structural changes to the system. [2] Rigidity and centralization based on a centralized and hierarchical approach bring the disadvantages of slow responses to abnormalities arising from the environment and the system itself as well as poor resistance to malfunctions [3].

On the other hand, systems that are based on purely heterarchic architectures (e.g., multi-agent systems [4]) are able to fulfil these requirements, as management units are intelligent entities whose cooperative strategies are based on autonomy, self-organization, and minimal global information. However, due to these properties, these systems are not deterministic. It is not possible to predict their future states [5], and thus it is not possible to achieve the level of system performance [6] that can be achieved with traditional hierarchical architectures during their faultless operation.

Another lack of heterarchic architectures is the absence of methodologies and standards that would directly address the structure of decision-making entities, cooperative methods, communications and interoperable protocols [7, 8]. Although the heterarchic architectures have many advantages, due to the shortcomings mentioned, they have rarely been implemented into the production environment, not to mention laboratory or test applications [8, 9]. Self-organization of entities in heterarchic architectures can lead to emerging behaviour.

Emerging behaviour is a phenomenon that occurs when a higher level of system behaviour changes, as a result of the interaction of entities that occur at lower levels. Java's emergent behaviour manifests itself if the properties of a whole are not the sum of the properties of its parts (the principle of superposition does not apply). Such indications can also be observed with multi-agent systems where the behaviour of an agent's organization does not correspond to the defined behaviour of either agent. In other words, it can be said that interactions of agents in an organization may produce properties that may not manifest by individual.

The effect of emergent behaviour may be positive, e.g. allows the management system to deal with a very complex and dynamic environment [33] and complex tasks [34]. On the other hand, the result may be negative, too with unexpected or unwanted manifestations that reduce the performance of the system or may damage it or its parts.

Holonic manufacturing systems represent a combination of hierarchic and heterarchic architectures and hence a semi-heterarchic (hybrid) architecture that is inspired by social or biological systems.

During the development of holonic production systems, a vision of a holon factory was created, but this concept has not yet been implemented [6]. This effort has resulted (1) in the promising IEC 61499 functional blocks that provide modularity and object-oriented features and easy reconfiguration [10] and (2) into the concept of a virtual enterprise that temporarily represents the partnerships created between independent enterprises in order to gain competitiveness, by sharing capabilities and resources [11]. Another concept that emerged from the Holon vision was a holonic agent [12].

Holons have a certain degree of autonomy and are able to cooperate with other holons, but they must also be subject to the limitations resulting from the hierarchy they are in.

This article brings an approach that combines the advantages of hierarchical and heterarchic architectures while overcoming their disadvantages. The approach includes a representation of the degree of autonomy and its possible application.

To model the holonic approach, we have opted for a multi-agent system with a communication application based on the Delegate MAS (D-MAS) [13]. In our application, AGV agents are coordinated by pheromone communications, which are subordinate to the intelligent product agent but have a degree of autonomy that is derived from the presence of system disturbances.

This article is organized as follows: Chapter 2 contains a problem definition followed by the Delegate MAS feature. Chapter 4 describes the design method that is verified in Chapter 5 of the case study. The results and future work are described in chapter 6 named Conclusion.

## 2 PROBLEM DEFINITION

The definition of the problem in this article is as follows:

a) How to combine the advantages of hierarchical (predictability of future states, optimization of overall performance) and heterarchic (autonomy, self-organization, fault tolerance) control architectures to overcome their disadvantages (weak resistance to system disturbances, non-modularity in hierarchical architectures, the inability to predict future states, the impossibility of optimizing the overall system, or the emergence of negative phenomena due to emergent behaviour in heterarchic architectures) into the semi-heterarchic (holon) governing architecture?
b) A dynamic degree of autonomy should determine / delimit the subordinate unit behaviour that may be on a scale that indicates the space between the heterarchy and the hierarchy. How to express, represent and apply such a dynamic degree of autonomy to subordinate AGVs in coordination and management? How can the dynamics move smoothly between the hierarchy and the heterarchy of subordinate units?

## 3 DELEGATE MAS

The concept of the delegated multi-agent system (D-MAS) was coined in [13]. It is a mechanism for coordinating and managing entities, inspired by food foraging behaviours in ant colonies. Application level agents (for example, a product, a production or logistic resource) delegate the role of their behavioural modules [14], called delegate MASs [15]. D-MAS includes lightweight agents (called ant agents or ants) that perform an assigned task; collect information and / or disseminate in the agent's intent environment. Aging agents typically have a limited calculation potential, memory and lifetime [16].

The ant agents use digital pheromones for indirect communication. A digital pheromone is a data structure that locally provides global (or remote) information [17]. This pheromone can formicate agents drop into the virtual environment, smell or modify it. Because pheromones are extinct in time, their intensity has to be periodically restored. Vaporization allows the system to forget about a solution that is inaccessible or worse than the new one found. This mechanism allows the system to automatically deal with disturbances and changes [18]. Forget-and-refresh mechanism ensures that the information in the environment is up-to-date [19]. D-MAS has been shown to be an effective coordination and control mechanism [20] and has been designed and used in a wide range of applications, from anticipatory vehicle routing [23], pick-up and delivery problem [24] dynamic service composition [25] to after multi-agent route planning [26].

Application level agents are divided into two types [15]:

a) Resource agent (e.g., drilling machine, conveyor) that provide their services / capabilities and functionality to the agent and
b) Task agent (e.g., product, mobile robotic system) that services consume.

The Resource Agent is responsible for answering the question of what-if, creating booking schedules, and maintaining up-to-date information on the services offered to its physical part and its qualities [17].

The Task Agent manages tasks in Coordinating and Control (C & C) applications. It typically represents a physical or virtual mobile entity. It has requirements that have to be met with respect to its objective [13].

The coordination mechanism makes it possible to take into account not only the current state of the system but also the state of the system in the near future [16] due to the propagation of intentions of task agents and thus can predict the future load of resource agents.

The antagents travel through the graph-structured network, whose nodes are resource agents. Edges in this chart topology represent links between agents. Each node contains a pheromone infrastructure that maintains and updates the intensity of the inserted pheromone. [15]

There are three types of D-MAS [15]:

c) feasibility,
d) exploration and
e) intention.

Feasibility ants are deployed by a resource agent and travel to the graph environment to extend feasibility pheromones of encountered resource agents and their services. Feasibility pheromone provides information on the availability of services / capacities. Feasibility ant

combines information from each resource agent into a feasibility pheromone that inserts into each node and road sign to the direction it came from.

Exploration ants are created at the location of its task agent. They traverse the graph environment and search for available solutions. Asking for resource agents asking what-if (e.g., what timing would be if a task agent would arrive at the particular time). Exploration ant then returns with the results back to its task agent. They can leave behind the digital roadsigns [22] that point to the already travelled path.

After the task agent selects one path through the graph environment, and it becomes its intention, it deploys intention ant agent to extend the information about that intention and make future allocations to each node on the intended path, thus informing the resource agent about the future visit by its task agent [16]. This feature allows you to generate short-term forecasts of the future load of agents and short-term forecasts of the agent's intentions. The above-mentioned approach is periodically repeated until the task agent meets its requirements. Finding and reserving / allocating path / services is usually called the refresh cycle and is based on a certain periodicity.

There are several research strategies, respectively walk roles [19] that can be applied to D-MASs when moving agents through graph environment. One is "propertinerary - itinerary of properties" [21]. This pattern maintains a list of destination properties, not destinations, and defines a routing scheme based on the sequence of locations in this list. In other words, instead of the task agent visiting the explicit destinations, the propertinerary describes what properties the destinations to be visited must have.

One way to incorporate the properties of hierarchical architectures into the D-MAS is inertia [15], which can also be understood as the commitment level [27]. It can be expressed by the value that the price of the new path found through the graph environment must be better than the original one. This means that the agents remain committed to their intention until the new solution is good enough to make the path change pay. On the other hand, a high level of inertia in the system increases the reliability of the forecast but can cause an inability to adapt to change [28].

Another property that affects the D-MAS characteristics is the boundary of space-time forecast. There is a distance in space and time to what antagents travel. The further the boundary of the forecast in time and space is shifted, the less reliable it is. On the other hand, the short prediction causes myopia.

If the resource allocation of a single resource agent is allocated to a single task agent at a certain time interval, and at the same time that the given task agent is assumed to be a task, it is possible to propagate constraints such as are, for example, time windows in which the capacities of a single resource agent are already allocated to previous resource agents in the graph-structured network. If the exploration ant agent finds (by asking what-if) that the capacity at the time that its agent arrives at the resource agent is already allocated, the exploration ant would no longer continue, but would return with steps (i.e. nodes) by which he came and promoted the unavailability of this capacity at a given time, but with respect for his shift.

Simple patterns of behaviour at D-MASs spill into emerging behaviour that is highly organized and effective in coordination and management, and at the same time robust against the uncertainty and complexity of the dynamic environment. [13]

## 4 DESIGN METHOD

In this chapter, we propose an approach to combine the advantages of hierarchical and heterarchic control architectures into the holonic management architecture, while expressing and applying the dynamic degree of autonomy of AGVs in their coordination and management by the Intelligent Product Agent (IPA).

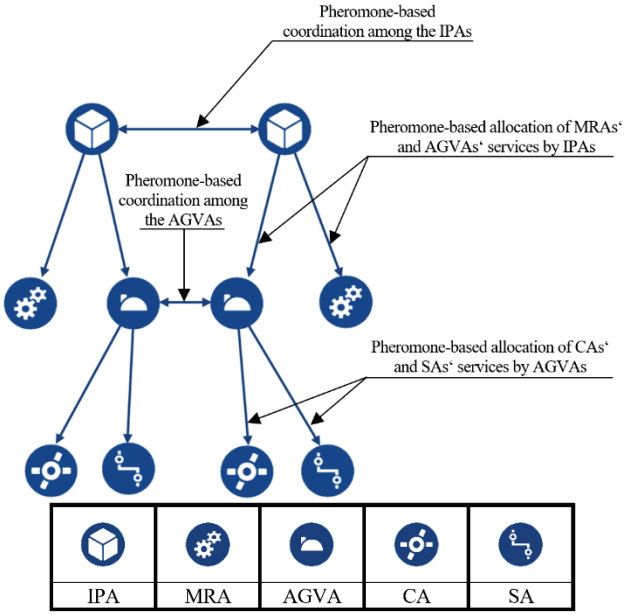


**Figure 1** Coordination based on stimergic allocations of services

### 4.1 CODESA-Prime

For the management and coordination of decentralized systems, the authors of this work have proposed the CODESA reference framework (COordination and COntrol of DEcentralized Systems Architecture). Application of this technology structure can be applied in domains where non-linear, heterarchic, wide-scale systems need to be managed in a dynamic, heterogeneous and unpredictable environment. CODESA-Prime is a specific application of the CODESA framework for the production domain. Using the proposed CODESA-Prime manufacturing control technology, it is possible to manage production processes from planning, scheduling, product routing, mobile agent routing to inventory management. Prime is characterized by 3 key agents: Intelligent Order Agent, Intelligent Product Agent, Intelligent Production Source Agent [29].

However, in this paper, for simplicity, we will focus on agent as resource or task agent. There are five types of agents in our approach:

a) Intelligent Product Agent (IPA),
b) Manufacturing Resource Agent (MRA),
c) AGV Agent (AGVA),

d) Crossroad Agent (CA) and
e) Segment Agent (SA).

Intelligent Product Agent encapsulates the features of the intelligent product [30, 31, 32]. It has a unique identity, it represents an instance of the product, it is capable of effective communication with the environment and uses the language to inform about its manufacturing and material requirements (in our case through pheromone-based communication). IPA can contain other Intelligent Product Agents (IPAs) and tasks in a recursive way.

Our approach is IPA task agent. It searches and allocates transport services to AGVAs and MRAs through exploration and intention agents. IPAs are coordinated based on allocated services via pheromones. Within a given timeframe, the service can only use one IPA.

Manufacturing Resource Agent (MRA) is a simple agent that can deploy feasibility ants to propagate their services, it can transform the intentional pheromones of the IPA into the schedule and is able to answer the what-if question. It is a resource agent type.

AGVA is similar to MRA but it is a mobile agent. In this case, this is a type of resume agent that offers services in the form of transport for IPA, and at the same time, it is a type of task agent who, through his exploration and intention ants, is looking for a crossroad and segment agent (Fig. 1). AGVA is coordinated and managed through ant agents. CA and SA represent parts of magnetic tape used to guide AGVA. Therefore, the graph-structure network at this level mirrors the magnetic tape. CA and SA have the types of resource agent and offer services in the form of a pass.

### 4.2 Intelligent Product Agent

IPA uses both types of communication, direct, for one-time message exchange, and indirect for repetitive actions and generating near-future forecasts.

A superior centralized unit in a hierarchical architecture usually represents a single point of failure while it is active throughout the production process. In our case, the IPA is a provisional unit only temporarily, and only if it uses pre-allocated services (Fig. 2). This increases the fault tolerance of the system. In the case of service allocation (MRA, AGVA) these agents become subordinate units but with a certain degree of autonomy (as parts of holon).

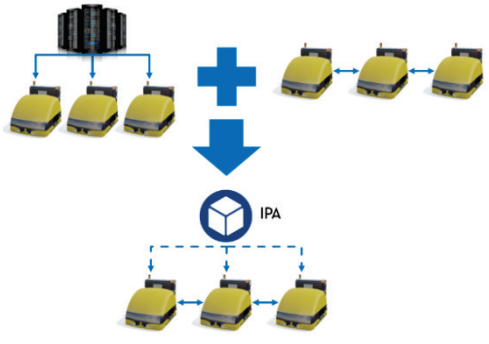


**Figure 2** Joining hierarchical and heterarchic architecture into the semi-heterarchical (holonic) control architecture

The dynamic degree of AGVA autonomy that can be changed during system operation is derived from the superiority of the IPA. In our case, the priority is based on two variables time remaining to due date of the IPA ($trbdd_{\mathrm{IPA}}$) and remaining processing time of the IPA ($rpt_{\mathrm{IPA}}$). Several assumptions about the priority were made.

Priority should be limited in the range from (0, 1). In such an interval is easier to handle. It should gradually increase and be derived from disturbances. In our case, it is derived from the time left to the date of the IPA ($trbdd_{\mathrm{IPA}}$) and remaining processing time of the IPA ($rpt_{\mathrm{IPA}}$) so that the system can also take account of delayed and rush orders.

If $trbdd_{\mathrm{IPA1}}$ is much larger than $rpt_{\mathrm{IPA}}$, the priority increment should be smaller. The function should be able to express critical point of the priority when:

$$rpt_{\mathrm{IPA}} = trbdd_{\mathrm{IPA}} \tag{1}$$

i.e. remaining reserve time to manufacture (in our case is negative):

$$rttm_{\mathrm{IPA}} = rpt_{\mathrm{IPA}} - trbdd_{\mathrm{IPA}} = 0 \tag{2}$$

Priority should also be defined beyond this point in order to give priority to orders that are longer. If orders are too late, the priority should rise slowly.

However:

$$rttm_{\mathrm{IPA}} \in R; -\infty < rttm_{\mathrm{IPA}} < \infty \tag{3}$$

and needs to be mapped into the interval (0,1). Thus:

$$p_{\mathrm{IPA}} = \frac{\alpha}{\alpha + e^{-\beta(rpt_{\mathrm{IPA}} - trbdd_{\mathrm{IPA}})}} = \{p_{\mathrm{IPA}} \in R;\ 0 < p_{\mathrm{IPA}} < 1\} \tag{4}$$

where: $p_{\mathrm{IPA}}$ – priority of the IPA, $\alpha$ – parameter for setting the critical point, $\beta$ – parameter for setting range of the function sensitivity.

We have applied the parameters for our simulation, where $rpt_{\mathrm{IPA}}$ is within [5-10] minutes and the average $trbdd_{\mathrm{IPA}}$ is from [0.15] minutes considering rush orders. The critical point was set to 0.8, leaving enough space to express the priority even after the feature reaches the critical point. The function is expressed as follows:

$$p_{\mathrm{IPA}} = \frac{4}{4 + e^{-\frac{1}{4}(rpt_{\mathrm{IPA}} - trbdd_{\mathrm{IPA}})}} = \{p_{\mathrm{IPA}} \in R;\ 0 < p_{\mathrm{IPA}} < 1\} \tag{5}$$

The resulting IPA priority function is shown in Fig. 3.

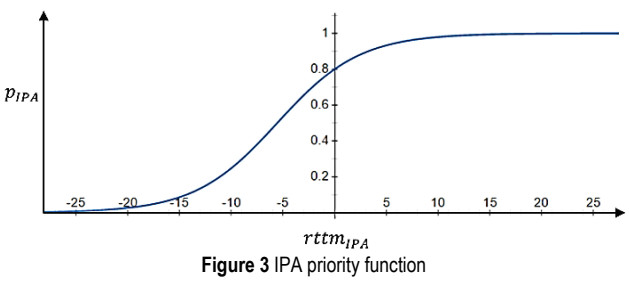


**Figure 3** IPA priority function

The IPA priority is applied when searching for and allocating services for MRAs and AGVAs. Each IPA begins its production in the warehouse and looks for the right (cheapest) production route through exploration ants. This is how the proprietary itinerary (propritinerary) is to be found, which in our case is a sequence of production and logistics operations. Exploration ants as the first operation look for transport from AGVAs, from the warehouse to MRAs, which can provide the next desired operation from the list. Subsequently, it is again looking for transport to further MRAs. In this case, the logistical requirements alternate with production. The illustration is shown in Fig. 4.

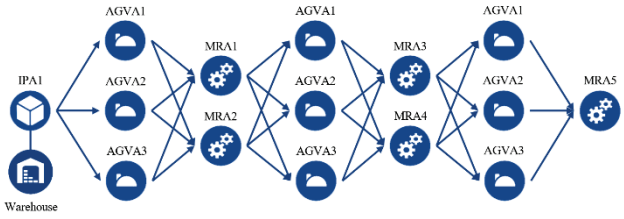


**Figure 4** Illustration of possible routes of the IPA

A demonstration of how the graphical environment, formed by the resource agents (AGVAs and MRAs), that travel exploration and intention ant agents, can be seen in Fig. 5.

The propagation of constraints (e.g., the allocation of service by MRA2 by IPA2 in time: [$t$ + 10, $t$ + 15]) back to warehouse is done by exploration ants of IPA. If the exploration ant gets a negative answer to the question "what-if", it means that the resource agent has its services at the given time already allocated. Exploration ant returns the resource agent, respecting the time shift (i.e. reflects the time that is required for the previous service) and inserts into the previous agent road sign pheromone, which points to the resource agent from where it comes, shifted the time of unavailability of this service, and the intensity of the pheromone that has faded over time, as well as the priority by which this service has been altered. In this way, it promotes constraints to the warehouse.

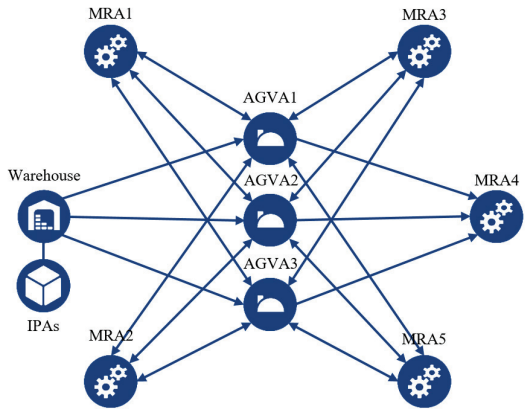


**Figure 5** Graph structured environment for IPA level

Propagated constraints run out of time to retest the given trip. If IPA acquires a higher priority than the specified constraint, it tries to allocate the routing for itself. If another IPA already actively uses the service for subordinate units, its priority for a given resource agent changes to an active and therefore non-numeric value:

$p_{\mathrm{IPA}} = MAX$, where:

$$p_{\mathrm{IPA}} = MAX > p_{\mathrm{IPA}} \in R;\ 0 < p_{\mathrm{IPA}} < 1 \tag{6}$$

Allocates services to subordinate units through intentional pheromones, which is dropped into the sub-unit's pheromone infrastructure. The intention pheromone dropped by the IPA's intention ants (IPIPA) provides the following information:

a) The unique name IPA, consisting of the name of the agent and the time stamp when the pheromone was created,
b) Start allocation (SDA),
c) End Date of Allocation (EDA), IPA priority,
d) Additional information (last refresh date, date of creation, etc.).

IPIPA, which is dropped into the pheromone infrastructure of the AGVA, also contains information about the transport order, thus:

a) from where, the starting position (unique name CA, or MRA)
b) where, destination (unique name CA, or MRA).

The IPIPA intensity is periodically renewed as well as its priority. If the IPA decides to change its direction, its intention pheromone will cease.

Usually, two IPIPAs are dropped for one transport order. The first to allocate services from the current location of AGVA to IPA, second, from the current IPA position to its destination. The exception is a transport order whose destination of the previous transport order is the starting position of the next transport order. The AGVA pheromone infrastructure has no maximum pheromones.

### 4.3 AGV Agent

AGVA derives its degree of autonomy from that of the current IPIPA. The higher the pheromone priority, the higher the degree of autonomy of AGMA. This degree of autonomy is reflected in the search for the path through CAs and SAs to the destination, similar to how IPA looks for its services in its subordinate units.

Since IPA is only a temporarily superior AGVA unit (only at the time of use of allocated services), the degree of AGVA autonomy ($da_{\mathrm{AGVA}}$) is also temporary and at the time it is equal to IPA priority level, respectively of IPIPA priority value:

$$da_{\mathrm{AGVA}} = p_{\mathrm{IPA}} \tag{7}$$

### 4.4 Application of Degree of Autonomy

We have proposed several ways to apply $da_{\mathrm{AGVA}}$. The first way to apply $da_{\mathrm{AGVA}}$ is the ability to reject IPIPA with a lower priority if it overlaps with an IPIPA priority interval with a higher priority while IPIPA with higher priority overrides the lower priority pheromone. In other words, AGVA is always trying to achieve the highest degree of autonomy.

$da_{\mathrm{AGVA}}$ may also appear when searching and booking a path through CAs and SAs. The higher the AGVA's

autonomy is, so the intention pheromones of its ant agentshave a higher priority. IPAGVA priority usage is the same as for IPIPA. Therefore, the higher priority takes precedence over the lower priority.

Inertial decision of AGVAs is represented by the inertial parameter ($ip_{\mathrm{AGVA}}$). The inertial parameter is a complement to AGVA's priority:

$$ip_{\mathrm{AGVA}} = 1 - da_{\mathrm{AGVA}} \tag{8}$$

Therefore, the higher the product's priority, the higher the $da_{\mathrm{AGVA}}$ and the smaller its inertial parameter. For incomplete orders or rush orders, the inertia is less (depending on the degree of autonomy), which makes it more likely to use a route with a lower price than its current one, so that it can find the fastest route to production. $ip_{\mathrm{AGVA}}$ represents a smooth transition between the hierarchy and the heterarchy. It is theoretically possible to say that if $ip_{\mathrm{AGVA}} = 0$, $ip_{\mathrm{AGVA}} = 1$ then AGVA is a fully autonomous system (heterarchy). If 1, AGVA is a fully subordinate system (hierarchy). In the latter case, the system is limited to single-shot optimization [35].

However, with low inertial parameters, the route is frequently changed and therefore the lubrication of the foreign directions of AGVAs. Therefore, the boundary of the space-time forecast is:

$$bstf_{\mathrm{AGVA}} = ip_{\mathrm{AGVA}} \tag{9}$$

The short forecast only affects the surrounding parts of the system. In this case, the forecast of the AGVA represents the number of allocated CAs and SAs.

### 4.5 Merge the Benefits of Hierarchical and Heterarchic Control Architectures

D-MAS lets you predict pheromones of IPAs and AGVAs, which allows you to optimize system performance. At the same time, the dynamic degree of autonomy applied to a given subordinate unit level is maintained. Self-organization allows you to find new solutions. D-MASs go into simple patterns that suppress the negative components of emergent behaviour. Exploration and intention ant colonies, forget-and-refresh mechanisms of the pheromones, increase resistance to system failures. The inertial parameter allows the system to converge to one solution and provides a smooth transition between the hierarchy and the heterarchy of subordinate units. The use of short-term forecasts of the future load of the resource agents allows forecasting of the state in the near future, as a partial optimization of the system as well as the suppression of negative phenomena from the emergent behaviour of self-organizing entities.

## 5 CASE STUDY

For dynamic verification of our solution, Ella Software Platform was used. Ella represents the virtual environment that originated at the University of Zilina and is further developed in the EdgeComcompany in Zilina. It is written in C ++.

The primary purpose of the Ella platform was to test robotic systems, but nowadays it provides modules from virtual reality through a virtual trainer to the ability to simulate mobile robotic systems. It provides a 3D graphical interface as well as a software physical library.

We used the modified Zilina Intelligent Manufacturing System (ZIMS) (Fig. 6). ZIMS is a platform that was created in cooperation with CEIT, and. s., Slovakia (Central European Institute of Technology) spin off the University of Zilina [36-41], Technical University of Košice, technological and industrial partners.

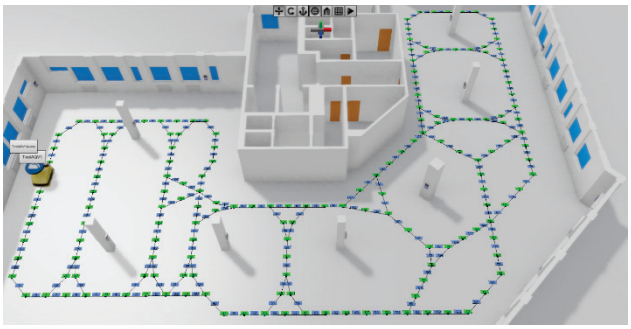

**Figure 6** Modified model of ZIMS in Ella Software Platform. Graph network infrastructure for AGVA C & C level. Black lines and nodes represent magnetic tape, thus CAs and SAs, which have a unique numerical identifier.

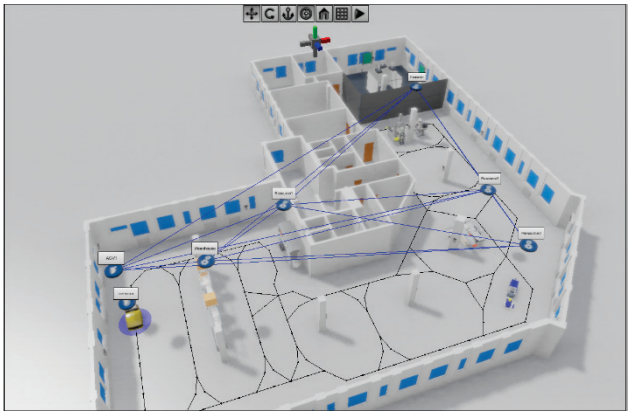

**Figure 7** AGVAs and MRAs form the graph-structured network for the IPA C & C level

We then implemented models of MRAs into the system. Each MRA can offer one operation (e.g., MRA1 provides Operation1, etc.), each with a specified duration of operation and cost. Subsequently, we created a graph network structure for the IPA C & C level (Fig. 7).

Testing the system has caused problems that we did not anticipate when designing it. Coordination and management at the AGVA C & C level limit the properties resulting from magnetic tape guidance, namely, the impossibility of rolling, the impossibility of rotating on the spot, or the inability to circumvent other unintentional AGVs. An autonomous logistics tractor can only move forward (change direction at junctions) or stop. In our case it means that the AGVA routing through CAs and SAs had to be allocated to its target node (MRA) and therefore the boundary of the space-time forecast could not be used for a certain number of CAs and SAs. There have been frequent dead-lock situations where AGVAs have been blocked in the opposite direction.

In order to learn how to set the boundary of space-time forecast, we have implemented a new type of agent we called Avatar. One agent may have multiple avatars. Avatar is created and managed by AGVA. Avatar, unlike an agent, has no decision-making ability but can create an ant colony and collect information from AGVA.

If an avatar finds and allocates routing to the AGVA C & C level and, in the case of other transport orders, at the end of its routing (the goal of one transport), by intention ant, will create another avatar, which will take the date of termination of the previous transport and set that date as the beginning of the next transport (whether IPA or only AGVA). In other words, there is one avatar for one IPIPA. In this way, it is possible to move the boundary of forecast further into the future.

The performance of the system was expressed as follows:

$$p_s = \frac{n_{cdd}}{n_{cdd} + n_{ncdd}} \cdot 100 \qquad (10)$$

where: $p_s$ – system performance, $n_{cdd}$ – the number of products produced by the defined date, $n_{ncdd}$ – number of products not produced until the defined date time.

Testing the system (Fig. 8) for fault-free operation and adding intelligent product agents with sufficient time before the date of manufacture was 100% system performance. The system was relatively well managed by addition of products whose date of production coincided with the time of routing through production and logistics sources. There have been occasions when the product has been delayed for a few seconds.

**Figure 8** Testing the system

To simulate abnormalities in the system, simulation of faults was implemented. The failure occurs with the probability $p_f = 0.0001$ in each calculation step, with the average elapsed time between the calculation steps being approximately 0.01667 seconds.

In the simulation with the probability of abnormal states (AGVA was a non-functional given number of seconds that was generated from the time interval [15-45] s.) and by adding agents of intelligent products with sufficient time reserve before the date of their production, the performance of the system was 100% that the system has managed its abnormalities well. When adding intelligent product agents to the edge (no time reserve), the production rate was approximately 60%.

## 5 CONCLUSION AND FUTURE WORK

In this article, we have proposed an approach that combines the advantages of hierarchical (predictive of future state and optimization of total system performance) and heterarchic (autonomy, self-organization, modularity, fault tolerance) architectures, to overcome their disadvantages in system disruptions, non-modularity in hierarchical architectures, and inability to predict future states, the impossibility of optimizing the overall system, the emergence of negative phenomena due to emergent behaviour in heterarchic architectures) into one semi-heterarchic (holon) governing architecture. This has been achieved by implementing approaches such as Intelligent Product, Pheromone-Based Communication (D-MAS) and foundation of holonic systems.

Emergent control architecture offers the possibility of a short-term forecast in the near future, and thus the possibility of planning or partial optimization of the system. The short-term prediction suppresses the negative phenomena resulting from the emerging behaviour of self-organizing entities, while the positive phenomena resulting from the emergence of a system of settlement with unforeseen circumstances and distortions remain unchanged. At the same time, it shares the benefits of modularity and resilience to heterarchic architecture failures, while the mechanism for "forget-and-refresh" based pheromones enhances, even more, resistance to system disturbances.

The degree of subordinate unit autonomy (in our case AGVAs) is dynamically adjustable during system operation. It allows for a smooth transition between the heterarchy and the hierarchy, its value falls within the interval [0,1] for better manipulation. We suggested possible approaches to this degree of autonomy in subordinate units, namely:

Ability to reject pheromones of the IPA with a lower priority than the degree of autonomy of AGV.

The expression of the intertial parameter, for a smooth transition between the hierarchy and the heterarchy.

However, as the verification has shown, due to the limitation of the movement of unidirectional AGVs, it is not possible to allocate individual resource agents in the AGVA routing, but it is necessary to allocate the entire path to the target. For this reason, we have created entities that can be used to set boundaries of the space-time forecast.

Higher priority of its own intentional pheromones, which are preferred over the intention of pheromones of lower priority (foreign AGVAs).

Simple verification at the end of the article confirmed the handling of simulated system disruptions.

### Acknowledgement

This paper was created with support of APVV project: APVV-14-0752: Reconfigurable logistic system for new generation of Factory of the Future manufacturing systems (RLS_FoF).

## 6 REFERENCES


[1] Leitao, P. et al. (2016). Smart Agents in Industrial Cyber–Physical Systems. *Proceedings of IEEE Conference on Control Applications, 104*(5), 1086-1101. https://doi.org/10.1109/JPROC.2016.2521931

[2] Van Brussel, H. et al. (1998). Reference architecture for holonic manufacturing systems: PROSA. *Computers in Industry, 37*(3), 255-274. https://doi.org/10.1016/S0166-3615(98)00102-X

[3] Wang, K. & CHoi, S. H. (2014). A holonic approach to flexible flow shop scheduling under stochastic processing times. *Computers & Operations Research*, 43, 157-168. https://doi.org/10.1016/j.cor.2013.09.013

[4] Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.

[5] Marik, V. et al. (2011). Automation's Holonic and Multiagent Control Systems Compendium. *Proceedings of IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and reviews, 26*, 90-96.

[6] Leitão, P. et al. (2013). Past, present, and future of industrial agent applications. *Proceedings of IEEE Transactions on Industrial Informatics, 9*(4). https://doi.org/10.1109/TII.2012.2222034

[7] Leitão, P. (2009). Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence, 48*(11), 979-991. https://doi.org/10.1016/j.engappai.2008.09.005

[8] Karnouskos, S. & Leitao, P. (2016). Key Contributing Factors to the Acceptance of Agents in Industrial Environments. *Proceedings of IEEE Transactions on Industrial Informatics*, 696-703.

[9] Trentesaux, D. (2009). Distributed control of production systems. *Engineering Applications of Artificial Intelligence, 22*(7), 971-978. https://doi.org/10.1016/j.engappai.2009.05.001

[10] Vyatkin, V. (2011). IEC 61499 as enabler of distributed and intelligent automation: State-of-the-Art review. *Proceedings of IEEE Transactions on Industrial Informatics, 7*(4), p. 768. https://doi.org/10.1109/TII.2011.2166785

[11] Camarinha-Matos, L. M. et al. (2009). Collaborative networked organizations – Concepts and practice in manufacturing enterprises. *Journal of Computers & Industrial Engineering, 57*(1), 46-60. https://doi.org/10.1016/j.cie.2008.11.024

[12] Rodriguez, S. A. (2005). From analysis to design of holonic multi-agent systems: A framework, methodological guidelines and applications. *Dissertation thesis*. http://www.fipa.org.

[13] Holvoet, T. & Valckenaers, P. (2006). Exploiting the Environment for Coordinating Agent Intentions. *Environ. Multi-Agent Syst. III*. Springer Berlin Heidelberg, Berlin, Heidelberg, 51-66.

[14] Maes, P. (1990). Situated agents can have goals. *Robotics and Autonomous Systems, 6*(1-2), 49-70. https://doi.org/10.1016/S0921-8890(05)80028-4

[15] Holvoet, T., Weyns, D., & Valckenaers, P. (2009). Patterns of delegate mas. *Self-Adaptive and Self-Organizing Systems*, 1-9. https://doi.org/10.1109/SASO.2009.31

[16] Van Belle, J. (2013). A Holonic Logistics Execution System for Cross-docking. *Dissertation thesis*. https://lirias.kuleuven.be/bitstream/123456789/416324/1/PhD_JanVanBelle.pdf.

[17] Weyns, D., Holvoet, T., & Helleboogh, A. (2007). Anticipatory Vehicle Routing using Delegate Multi-Agent Systems. *Proceedings of IEEE Intell. Transp. Syst.*, 87-93. https://doi.org/10.1109/ITSC.2007.4357809

[18] Valckenaers, P., Saint Germain, B., Verstraete, P., & Van Brussel, H. (2007). MAS coordination and control based on stigmergy. *Computers in Industry, 58*(7), 621-629. https://doi.org/10.1016/j.compind.2007.05.003

[19] Valckenaers, P. & Van Brussel, H. (2005). Holonic manufacturing execution systems. *CIRP Annals-Manufacturing Technology, 54*(1), 427-432. https://doi.org/10.1016/S0007-8506(07)60137-1

[20] Hanif, S. & Holvoet, T. (2014). Dynamic Scheduling of Ready Mixed Concrete Delivery Problem Using Delegate MAS. *PAAMS*, 146-158. https://doi.org/10.1007/978-3-319-07551-8_13

[21] Holvoet, T., Weyns, D., & Valckenaers, P. (2010). Delegate MAS patterns for large-scale distributed coordination and control applications. *Proceedings of the 15th European Conference on Pattern Languages of Programs*, p. 25. https://doi.org/10.1145/2328909.2328940

[22] Philips, J., et al. (2013). Computational Complexity and Scalability Analysis of PROSA and delegate MAS. *IFAC Proceedings Volumes, 46*(7), 29-34. https://doi.org/10.3182/20130522-3-BR-4036.00020

[23] Claes, R., Holvoet, T., & Weyns, D. (2011). A decentralized approach for anticipatory vehicle routing using delegate multiagent systems. *Proceeding of IEEE Transactions on Intelligent Transportation Systems, 12*(2), 364-373. https://doi.org/10.1109/TITS.2011.2105867

[24] Hanif, S., van Lon, R. R., Gui, N., & Holvoet, T. (2011). Delegate MAS for large scale and dynamic PDP: A case study. *Intelligent Distributed Computing V*. Springer Berlin Heidelberg, 3-33. https://doi.org/10.1007/978-3-642-24013-3_4

[25] Torres, M. H. C. & Holvoet, T. (2014). Self-adaptive resilient service composition. *Proceeding of IEEE Cloud and Autonomic Computing*, 141-150. https://doi.org/10.1109/ICCAC.2014.33

[26] Dinh, H. T., van Lon, R., Holvoet, T. (2016). Multi-agent route planning using delegate MAS. *Workshop on Distributed and Multi-Agent Planning, T.*, 24-32.

[27] Torres, M. H. C., & Holvoet, T. (2011). Towards robust service workflows: a decentralized approach. *OTM Confederated International Conferences on the Move to Meaningful Internet Systems*. Springer Berlin Heidelberg, 155-162.

[28] Valckenaers, P., Verstraete, P., Saint Germain, B., & Van Brussel, H. (2005). A study of system nervousness in multi-agent manufacturing control system. *International Workshop on Engineering Self-Organising Applications*. Springer Berlin Heidelberg, 232-243.

[29] Durica, L., et al. (2015). Manufacturing Multi-agent system with bio-inspired techniques: CODESA-Prime. *MM science journal* / Prague. https://doi.org/10.17973/MMSJ.2015_12_201543

[30] McFarlane, D., Sarma, S., Chirn, J. L., et al. (2003). Auto ID systems and intelligent manufacturing control. *Eng Appl Artif Intell, 16*, 365-376. https://doi.org/10.1016/S0952-1976(03)00077-0

[31] Karkkainen M., Holmstrom J., Framling K., & Artto K. (2003). Intelligent products: a step towards a more effective project delivery chain. *Computers in Industry, 50*(2), 141-151. https://doi.org/10.1016/S0166-3615(02)00116-1

[32] Venta, O. (2007). Intelligent Products and Systems Technology theme - Final report. Helsinki.

[33] Steegmans, E., et al. (2002). Ant algorithms in a graph environment: a meta-scheme for coordination and control. *Memory*.

[34] Tatomir, B. & Rothkrantz, L. (2006). Hierarchical routing in traffic using swarm-intelligence. *Proceeding of IEEE Intelligent Transportation Systems Conference*, 230-235. https://doi.org/10.1109/ITSC.2006.1706747

[35] Van Belle, J., Philips, J., Ali, O., Saint Germain, B., Van Brussel, H., & Valckenaers, P. (2012). A service-oriented approach for holonic manufacturing control and beyond.

*Service Orientation in Holonic and Multi-Agent Manufacturing Control*, Springer, Berlin Heidelberg, 1-20.
[36] Dulina, L. & Bartanusova, M. (2014). Ergonomics in practice and its influence on employees' performance. *Communications: scientific letters of the University of Zilina, 16*(3A), 206-210.
[37] Micieta, B. et al. (2014). Delegate MASs for coordination and control of one-directional AGV systems: a proof-of-concept. *Advanced Manufacturing Technology, 16*(3A), 101-106.
[38] Micietova, A. et al. (2014). Study of Elastic Deformations in Hard Turning. *Key Engineering Materials*, 581, 176-181.
[39] Kohar, R. & Hrcek, S. (2014). Dynamic analysis of a rolling bearing cage with respect to the elastic properties of the cage for the axial and radial load cases. *Communications: scientific letters of the University of Zilina, 6*(3A), 74-81.
[40] Krajcovic, M. et al. (2013). Intelligent manufacturing systems in concept of digital factory. *Communications: scientific letters of the University of Zilina*, 15, 77-87.
[41] Rakyta, M. & Bubenik, P. (2014). Data mining technology and its benefits in business practice. *Manufacturing systems today and tomorrow 2014. Proceedings of the 8th annual international Conference* / Liberec, 6-10.

**Contact information:**

**Branislav MICIETA,** Prof. Ing. PhD
University of Zilina
Univerzitna 1, 010 26 Zilina, Slovakia
E-mail: branislav.micieta@fstroj.uniza.sk

**Lukas DURICA,** Ing., PhD
University of Zilina
Univerzitna 1, 010 26 Zilina, Slovakia
E-mail: lukas.durica@fstroj.uniza.sk

**Vladimira BINASOVA,** Ing. PhD, DiS.
University of Zilina
Univerzitna 1, 010 26 Zilina, Slovakia
E-mail: vladimira.binasova@fstroj.uniza.sk