Open manufacturing control with agile reconfiguring of resource services

Th. Borangiu*, **S. Răileanu***, **D. Trentesaux****, **T. Berger****

** University Politehnica of Bucharest, Dept. of Automatic Control & Industrial Informatics, Bucharest, Romania (Tel: 40/21 402 93 14; e-mail: borangiu@cimr.pub.ro, silviu@cimr.pub.ro)*
*** University Lille Nord of France, F-59000 Lille, TEMPO Laboratory, Valenciennes, France (Tel: 33/327511463; e-mail: Damien.Trentesaux@univ-valenciennes.fr, Thierry.Berger@univ-valenciennes.fr)*

Abstract: The paper describes a new, open control paradigm and implementing solution for discrete, repetitive shop floor production, designed as a frame for agile manufacturing reengineering through (a) multi-agent resource service reconfiguring and (b) implementing the robot service access model (RSAM) in a distributed, semi-heterarchical production planning, scheduling and execution control architecture. This distributed architecture integrates two layers with generic functionalities: (1) Dynamic reconfiguring of the resource (robots, vision systems, CNC machine tools) service access model RSAM through a multi-agent system organization; (2) Holonic manufacturing scheduling (planning, resource allocation), control and tracking based on the PROSA frame and the Intelligent Product technology – implemented through intelligent embedded devices (acting as Active Holon Entities) which use OpenEmbedded Linux as real-time operating system. Experimental results are reported from production scenarios tested in the two shop-floor platforms of the Laboratories AIP-PRIMECA (University of Valenciennes) and CIMR (University Politehnica of Bucharest) composed by robot-vision and machine tool workstations.

Keywords: agile manufacturing, holonic control, batch scheduling, intelligent product, embedded devices

1. INTRODUCTION

Current advances in information technology and electronics made possible attaching devices with decisional and communicational capabilities to almost all of the entities of a flexible manufacturing system (FMS). This allows passing from the classical centralized control approach to a fully decentralized and more flexible control approach where each entity (e.g. product, robot, machine tool resource) has its own objectives, making it however difficult for the system as a whole to achieve a global objective (like minimal production time or balanced resource load at batch level). Despite the structural differences between these two control architectures, common aspects still exist, and only the way in which they are treated differs. Such generic aspects are *batch production planning* and *resource allocation* (*scheduling*). To further optimize the production, a combination of the two problems has been proposed in (Barták, 2000; Babiceanu et al., 2004), but most of the classes of algorithms (of operational research or applied artificial intelligence type) offer good results only for cvasi-deterministic environments and induce certain costs which are mainly determined by their runtime necessary to offer a real-time solution during batch execution control.

Since reality is rarely so deterministic, centralized approaches rapidly become inefficient when the target system must deal with disturbances or uncertainties relative to resources and material flows, which may switch the primary objective of a designed system from global optimization to fault tolerance at resource breakdowns and agility at client demands (e.g. rush orders, Sauer, 2008). This led researchers to define new approaches to designing shop-floor control architectures that self-organize the access to their resources to feature agility to high-frequency production changes, adaptability to material flow variations and efficiency in resource utilization.

Such advances change also the product scheduling problem which is done now in coordination by several information entities instead of being done centralized by a single entity (Murillo et al., 2009). Based on these guidelines, new research directions in manufacturing control have been proposed, which are centred on *product-driven* scheduling, batch execution and tracking. This control paradigm is based on the "intelligent product" concept and implementing frame (McFarlane D., 2002), and a complete survey on this field was done by (Meyer et al., 2008) who classifies intelligent product solutions according to three axes: level-, location- and aggregation- of intelligence.

The above observations have motivated researchers to design emergent or self-organized control architectures, which fall in two categories: (1) **Multi Agent Systems** (MAS) for *agile reconfiguring of resource service access model*, as proposed by (Mayone et al., 2003) and (Barata, 2006); (2) **Holonic Manufacturing Execution Systems** (HMES) for *production planning, scheduling, control and tracking*, as the: reference architectures proposed by (Van Brussel et al., 1998; Leitao, 2006), implementing frame proposed by (Chen et al., 2006), semi-heterarchical model defined by (Babiceanu et al., 2004) and implementing frame proposed by (Borangiu et al., 2009), and the heterarchical model proposed by (Trentesaux, 2009) who introduces a global control paradigm called "open-control" in which traditional control is augmented by a new kind of control: "implicit". In this paradigm, entities can be strictly controlled hierarchically and be at the same time influenced heterarchically by their environment and/or by other entities (Trentesaux, 2007).

2. AGILE RECONFIGURING OF RESOURCE SERVICES AND OPEN CONTROL MECHANISMS

A shop-floor control framework is proposed in which one entity (e.g. a processing, transport or inspection resource, a product) can not only achieve its goal in terms of the system's objectives but also in terms of its own objectives. An entity can be a *resource* (e.g. a machine, a robot, a vision system) or an active *product*. An **active product** or Active Holon Entity (AHE) is an aggregate entity able to inform, communicate, decide and act in order to reach its goals in solving resource allocation and routing problems (Fig. 1).

The target repetitive, discrete batch fabrication structure consists of several machine tool-, robot-, vision- and storage-workstations interconnected by a shop-floor conveyor. Each workstation contains one or more processing resources (CNC machines), a part handling & processing robot (accessing the cell conveyor) and product control unit (machine vision).

The products, which are placed on **pallets**, are progressively processed and assembled by physical resources, some of which are identical or have identical capabilities; some of the resources are different but offer similar services at different costs. Each pallet mounted on a carrier moving on the conveyor is equipped with an Intelligent Embedded Device (IED) which is capable of memorizing information, communicating over an ad-hoc network with peer devices and taking real-time decisions regarding *product scheduling* (allocating a resource to each operation on products), *product and resource tracking* (monitoring the operation's quality and the resource's performance, creating the product's "history").

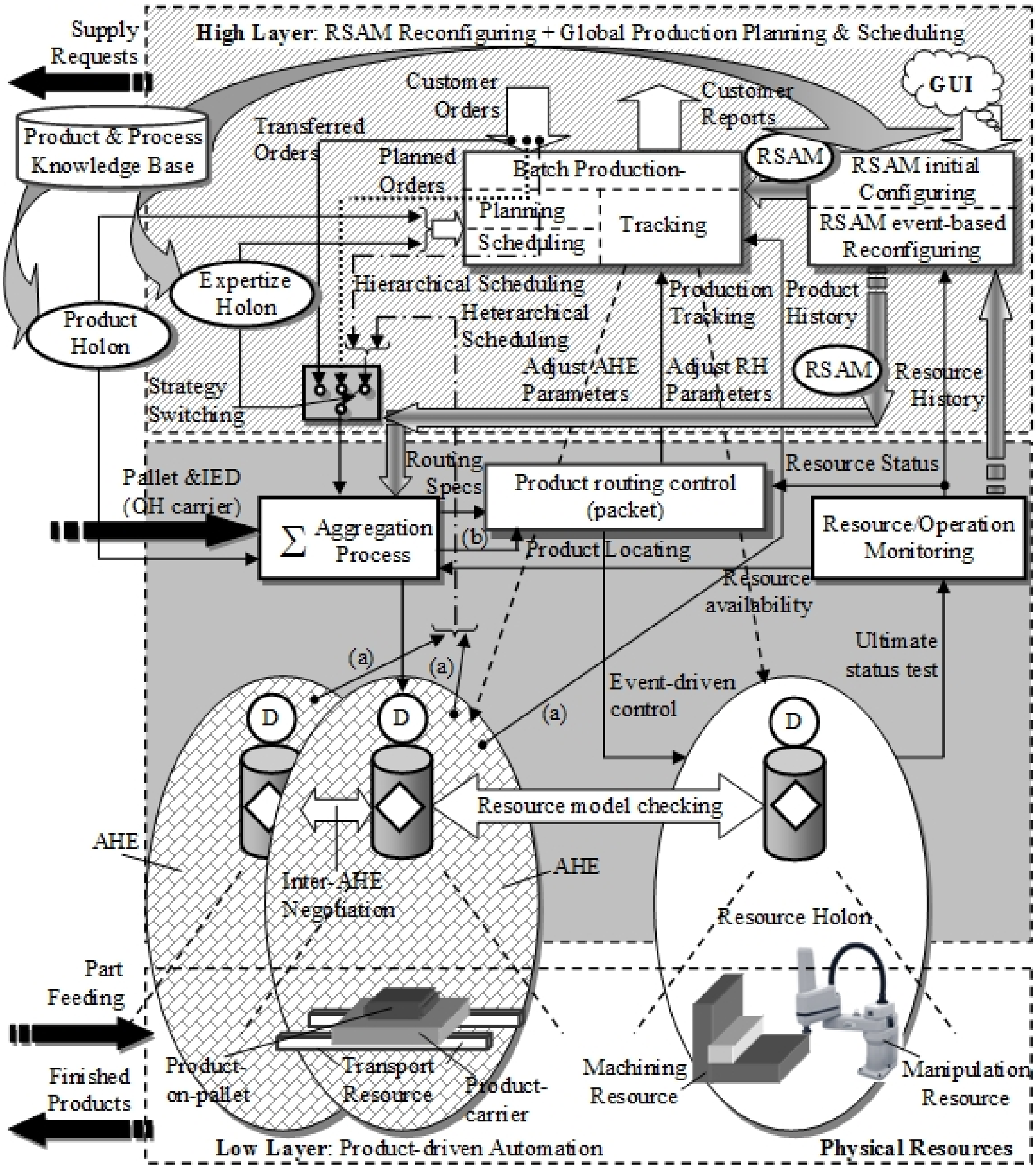

Fig. 1. The two-layer generic architecture for dynamic (re)configuring the Resource Service Access Model (RSAM) and open, semi-heterarchic shop-floor control (batch production planning, scheduling, controlling and tracking)

The proposed control architecture, called "open-control", has the advantage of augmenting traditional explicit control with a new kind of control - "implicit control". In this paradigm, entities can be strictly controlled hierarchically and, at the same time, they can be influenced in a heterarhical mode by the environment in which they operate ("environmental control") and / or by other entities ("societal control"). This feature allows designing a control system which is both agile and globally optimized, thus reducing the myopic behaviour of self-organized architectures and increasing the agility of traditional architectures.

Combining the two types of control: explicit (for optimality at batch level) and implicit (for agility and fault-tolerance) in the same architecture creates new challenges since the two types of control must now be managed and integrated within the larger control paradigm. The components of the implicit control are handled as follows:

- *Societal implicit control.* This type of implicit control is performed in two ways. (1) The first involves <u>fine tuning the partial view of a collective property</u> inside an entity representing the *service sequence* and *providers* (set of the physical resources) to manufacture a product. This modification can be seen as an internal influence that modifies the entity's behaviour. This behavioural modification then influences the other entities via the societal optimization mechanism, which is supported by **dialogue**. For example, a controller can force a specific product type to be machined on a specific resource, which implies changing the dynamic of the allocation process for the other products. (2) The second way involves <u>changing the dynamics of the dialogue</u> in the societal optimization mechanism by modifying the dialogue parameters of these entities (Active Holon Entities that evolve in Holon Orders after real-time scheduling within the product-driven automation (low) layer of the system). For example, in a contract-net context, a product can interrogate all the resources or only those resources in its proximity. This second way has a direct impact on the overall collective performance.

- *Environmental implicit control.* This type of implicit control is performed via the informational environment in two ways: the first determines <u>acting on the data directly</u> (e.g., creating, updating, erasing data files, records), while the second one involves <u>fine tuning the parameters</u> used by the environmental optimization mechanism (e.g. create task-driven virtual cameras for quality control, authorise robot access to parts by real-time clear [collision-free] grasp check, influence robotic part handling by visual part qualification). Both actions generate external influences that can affect all entities able to access the informational environment. For this type of implicit control, sensor reading and data fusion is necessary; no communication between entities is required.

In implicit control, the final entity is not directly targeted. Implicit control uses a dedicated intermediate, which does not directly target the influenced entities, but rather adjusts the properties and / or behaviour of entities (e.g. resources).

In a manufacturing system composed of autonomous entities, each entity is immersed in an informational level orchestrated by an **Optimization Mechanism** (OM), and each entity is always trying to achieve its own objective through a decision-making process that is influenced by either a societal or an environmental OM (see Fig. 2). Each entity is free to achieve its own objective, but in terms of global system performance criteria that apply to all entities. In general, the system must be used by all entities and must work for all entities.

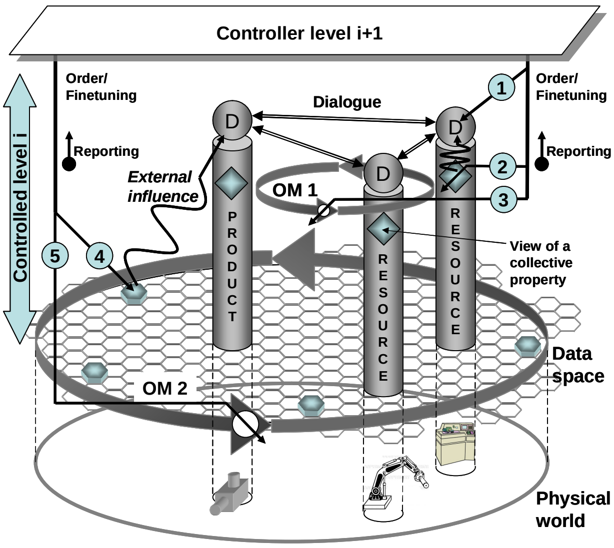

Fig. 2. OMs in open manufacturing control (1 – explicit; 2,3 – societal implicit; 4, 5 – environmental implicit)

In Fig. 3, the two main levels performing product-driven automation / RSAM configuring and global production planning and scheduling (at batch level) are respectively defined as "controlled level *i*" (low layer in Fig. 1) in which the different entities evolve, and "controller level *i*+1" which controls level i. The physical world is also represented through the physical base of the different entities.

As mentioned above, each entity must meet its own goal but in terms of collective global performance criteria. At their individual levels, the entities all have a self-made view (personal knowledge) of the collective performance criteria (represented in Fig. 2 by a small diamond). This partial view is achieved by dialogue among entities. For example, in a classic contract-net approach, an active product can dialogue with the resources and thus obtain a view of their availability and choose the most appropriate. These exchanges support a mechanism for optimizing collective performance.

OM1 in Fig. 2 is called *societal OM*, because it only concerns the entities and not directly their environment. The entities also have access to an informational environment, composed of data spaces placed in certain locations (represented by small hexagons in Fig. 1). The entities can access via sensors the information available in their vicinity and integrate this information into their decision-making. The entities can also enrich the data spaces with their own experiences, which may correspond to a collective performance criterion. For example, in transport tasks, marks located in diverting points can be used to provide a view of overall traffic fluidity; the entities' travel experiences can be used to update these marks.

The informational environment (labeled data space in Fig. 2) is dynamic and governed by positive and negative feedback. These two types of feedback compose the second type (OM2) of optimization mechanism, called *environmental OM*.

Because **agility** is a main objective to be achieved by the open control structure, a generic multi-agent architecture is proposed to allow shop-floor reengineering; components of this architecture and their capabilities are configured initially (upon receiving customer's orders) and possibly reconfigured at run time in case disturbances occur (adding / removing one resource, performance decrease or breakdown of a resource). A **Resource Service Access model** (RSAM) is thus created and maintained using generic properties:

- *Modularity*: a production system will be configured as a dynamic composition of modularized manufacturing units which become basic building blocks. Building blocks are developed on the basis of processes they are to cater for.
- *Configuring rather than programming*: the addition or removal of any manufacturing component (basic building block) should be done smoothly, without or with minimal programming effort. The system composition and its behaviour are established by configuring the relationships among modules, using contractual mechanisms.
- *High reusability*: the building blocks should be reused for as long as possible, and easily updated for further reuse.
- *Legacy systems migration*: legacy and heterogeneous controllers might be considered in any global architecture and a process must be developed to integrate them in the new agile architecture.

Two stages were considered in the creation of the shop-floor Resource Service Access Model:

1. Initial creation (*RSAM configuring*): using a Graphical User Interface (GUI in Fig. 1), resources are manually added to the working structure or team (responsible for producing a type of product), being thus created a map of services offered by the team, their costs and the way they can be accessed.
2. Automatic update of resource status at run time (*RSAM reconfiguring*): the resources are monitored by the Active Holon Entities during their lifecycle and the resource access model is updated with information about the real-time capacities of each resource, its availability and the penalty/bonus it received for the accomplished services.

Shop floor agile control / supervision can be achieved if the manufacturing system is abstracted as a composition of modularized manufacturing components that can be reused whenever necessary, and whose interactions are specified using reconfiguration rather than reprogramming. Consequently, a generic multi-agent system (MAS) was designed to create and automatically update the shop floor's service access model RSAM, because of its adequacy to create cooperative environments of heterogeneous entities.

Manufacturing components were *agentified* to become modules that can be (re)used to compose complex systems. The different types of manufacturing scenarios and batches were thus represented by coalitions or consortia of agentified manufacturing components, which are essentially societies of self-interested and heterogeneous agents whose behaviour is governed by contracts (Barata, 2006); contract negotiation is the configuration basis required whenever a supervision / control system needs to be changed or adapted.

Thus, a manufacturing component or module was seen as a physical piece of equipment that can perform a set of specific functions / basic production actions on the shop floor such as moving, transforming, handling or inspecting. To design the generic RSAM, images of the manufacturing were defined as:

- *Agentified manufacturing component*: composed of a manufacturing component and the agent that represents it. The agent's skills are those offered by the manufacturing component, connected to the agent through middleware.
- *Coalition* or *consortium*: a group of agentified manufacturing components, whose cooperation is regulated by a coalition *contract*, interacting in order to generate aggregated functionalities that, in some cases, are more complex than the simple addition of their individual capabilities.
- *Shop floor cluster*: a group of agentified manufacturing components which can participate in coalitions and share some relationships (belonging to the same manufacturing structure and possessing some form of technological compatibility). The different coalitions that can be created out of a cluster represent the different ways of exploiting / operating a manufacturing system.
- *Broker agent*: used to help the formation of coalitions to reduce the complexity of the individual agents in terms of coalition formation.

Once configured and operational the service access model of manufacturing resources, it was integrated into the 2-layer holonic control architecture for both high-level optimal batch products planning and scheduling and low-level real-time packet products (products currently in execution) scheduling. This architecture is intended for shop floor environments affected by disturbances like: resource unavailability due to breakdowns or maintenance operations, part stocks depletion due to limited storages, variable processing and transporting times. The proposed planning, scheduling and control models with their implementation frame are generic; the structuring of the decisional entities (Active Entity Holons) and the distributed decision making (based on holon autonomy and cooperation) do not rely on proprietary technologies.

The Holonic Manufacturing Execution System automatically switches between the global "batch" planning and scheduling horizon and the local "packet" scheduling horizon for shop floor resource assignment – thus providing user definable combined optimality, agility and fault-tolerance in business-oriented scenarios. Intelligent embedded devices (IED) assist products during their real-time scheduling, routing and tracking – thus bringing closer the physical and decisional parts of active entities (products and related execution data) performing their own objectives in a global batch context.

The control part of the distributed system is composed of entities that are independently responsible for one aspect of fabrication such as technological planning (**product** recipe), **resource** capabilities, and logistics (production **order**). These components, being endowed with information processing skills (except for products) are encapsulated into autonomous and communicative entities called *holons*. The following holons were defined:

1. On the high level layer:
   - A set of **Expertise Holons** (**EH**): together with the application for global production planning, scheduling and tracking acts as a *Coordinator* of the high level control with its attributes, including the client interfaces.
2. On the low level layer:
   - A set of identical **Active Holon Entities (AHE)**: one AHE is an aggregate intelligent entity in charge of taking real-time decisions. It is composed of: (a) the product being fabricated, (b) the pallet carrier which transports it to the assigned (scheduled) workstations where operations are executed upon according to the product recipe, and (c) an augmentation module, implemented as an *Intelligent Embedded Device* (IED) providing decisional capabilities. The maximal number of AHEs in the shop floor equals the dimension of the product packet in current execution.
   - A set of **Resource Holons (RH)**: describe the physical resources (e.g. robot, conveyor), used for part processing or transporting, together with their controllers and sensors which communicate with the AHE for service granting and management.
   - The **Product Holons** (**PH**) store the operations structure for all the types of ordered products, by retrieving info from a **Product** and **Process Knowledge Base** (PPKB).

In heterarchical production control mode, the AHE use the data from the PH to access, through communication with the RH, those available and cost-effective resources the image of which is permanently updated in the RSAM.

The generic model for production planning, scheduling, control and traceability is organized on two decision layers, with *semi-heterarhical* operating capability:

1. A **high level layer** in charge with collecting the clients' orders and performing the off-line decisional process of *long term planning* and *scheduling* (at batch horizon). The layer is connected to the user through an interface for order reception, reporting and RSAM configuring. The client's requests are first mapped to an Aggregate Product Orders list (APO) from the PPKB which also generates the list of operations describing the services to obtain for the execution of each product type. The APO is input to a centralized application which, using Expertise Holons, generates in hierarchical mode the list of optimally ordered and scheduled production orders which are then associated to the AHE at run time through a process of aggregation. The optimization of product scheduling is relative to a global cost function, at batch horizon, such as: *makespan*, *resource loading*, a.o.;
2. A **low level layer** in charge with *process automation* (AHE execution), i.e. with *implementing the production schedule* recommended by the high layer. This layer may switch its operational mode on request or automatically from *optimal long-term scheduling*, following one Expertise Holon strategy (at batch level), to distributed decision for ***short-term, on-line scheduling*** (at packet horizon – for the products currently in execution) in order to react at disturbances: resource failure, bottlenecks or new available paths on the conveyor (due to last moment changes in certain AHE's schedules). Upon switching in this heterarchical mode, a new type of **product-driven automation** is initiated meaning that *real-time scheduling* is done by the Intelligent Embedded Devices (IED) which are placed on the product carriers (pallets) rending them active as Active Holon Entities (AHEs).

Whereas the low control layer is in charge with implementing the heterarchical operation model when necessary (requested by the user or automatically triggered by the above described perturbation events), the high control layer is in charge with further choosing an adequate heterarchical strategy based on: (a) existing supervisor strategies; (b) current status and past behaviour of resources; (c) product execution history; (d) the current traffic on the transportation system; (e) the number of products not yet in execution. The information in (a)-(e) is collected, processed and transferred by the IED to the central entity in charge of strategy switching (the Expertise Holon).

3. ACTIVE HOLON ENTITY AND PRODUCT-DRIVEN SCHEDULING

The IED design approaches two essential problems in real-time, product-driven manufacturing control: (i) *product locating* and (ii) *decision making for resource allocation.* Both problems are influenced by the placement of the information part of the active holon entities (taking decisions in the process of real time scheduling) with respect to the physical part (the product carrier) and by the synchronization solution between the two parts.

A. Intelligent Embedded Devices for AHE

In the present design, the augmentation module associated to the AHE comprises (see Fig. 3):

- A *data storage module* memorizing the fabrication model of the product (operations to be done, their parameters and precedence between these operations – services to be obtained) and of the resource model (services provided by each resource and their costs, the current status of the resources and of the links between them – RSAM);
- A *module for communication* and
- A *decision module* (for real-time scheduling).

Two heterarchical scheduling strategies are proposed:

[1] *Local (sequential) optimization*: for all the products in execution (when all AHEs communicate and synchronize their schedules);

[2] *Next service search*: in this mode, an AHE travelling on the shop-floor transportation system identifies all free resources capable to offer the next requested service and chooses the nearest one without disturbing the other AHE carriers; if no free resource exists, it is re-circulated.

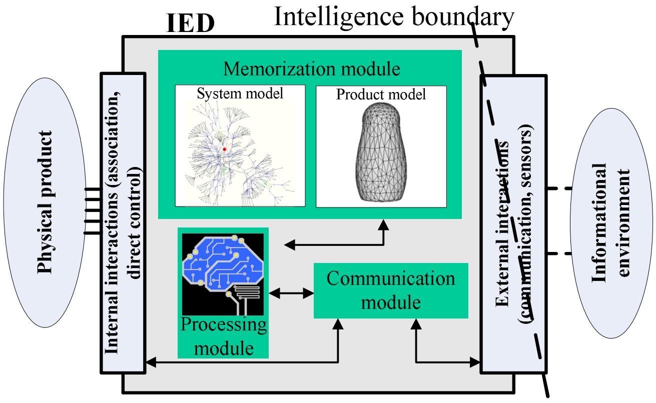

Fig. 3. Structure of the intelligent embedded device (IED) augmenting the OH with active behaviour to an AHE

The augmenting entity is thus structured as **local (embedded) intelligence**, placed on the product carrier, which rends the AHE entity more autonomous and co operant. The decision-making process is more agile since decision is taken near the point of interest, and more fault tolerant because in the case of a local failure the rest of the entities can continue to work. The entities do not rely in this case on the communication of control information but on the synchronization between them. Product localization is done in this case by the IED which interprets the signals received from sensors placed on the conveyor in the proximity of resources. This solution allows implementing "product-driven automation" (or "intelligent product" method), providing *agility* in operation and *modular structure* easy to change *by reconfiguration*.

The IED augmentation module monitors the system's status; the following events trigger the commutation process, strictly from long-term hierarchical optimized operation mode to short term heterarchical operation mode, characterized by agile resource re-allocation:

- If a resource breaks down and an AHE has operations allocated on it, it will need to reschedule these operations;
- If there is a resource that can execute an operation faster than the current scheduled resource (which performs a task much later than expected), than the newly discovered workstation will be used. This decision is taken based on the current location of the AHE and on the system status model updated with the most recent information (resource states, intervals with resources reservations by other products, transportation times);
- If there is a jamming on a conveyor segment, the AHE must initiate a rescheduling process, trying to clear the transportation path – a critical resource (the path to a resource from a certain point forward is usually unique);
- If a resource recovers from breakdown, scheduling at packet level and for the rest of products will be done.

B. Heterarchical Robot Service Allocation

The decentralized real-time resource allocation mechanism, presented in Fig.4, contains two key elements: a *local decision making agent* in charge with production monitoring, operation execution and resource allocation and a *mediator agent* in charge with conflict resolution. On each AHE both agents coexist: the decision making agent is active for each AHE while the mediator agent is active for only one AHE in the whole system. The mediator is elected dynamically, after the current one leaves the system, based on the rule "the oldest AHE in production will be elected mediator".

The following steps are taken in order to select the mediator:

STEP 1. All AHE exchange advertisements (thin lines in Fig. 3) between them containing their ID and uptime;

STEP 2. Each AHE selects a mediator based on the uptime (since there is only an input point all the uptimes will be different and an unique mediator will be selected);

STEP 3. All AHE will register (thick lines in Fig. 3) to the chosen mediator to solve conflicts.

In this architecture, a decision (*online scheduling** for all operations of an AHE) is taken based on the restrictions of the shop-floor, on the transportation and processing times and also trying to respect the production rule "the older AHEs have priority" (Fig. 5). All the information about the shop-floor structure is found in a copy of the *resource service access model* RSAM that is located on each AHE. This copy is updated in real-time with the real image located on the high control layer where resource monitoring is done.

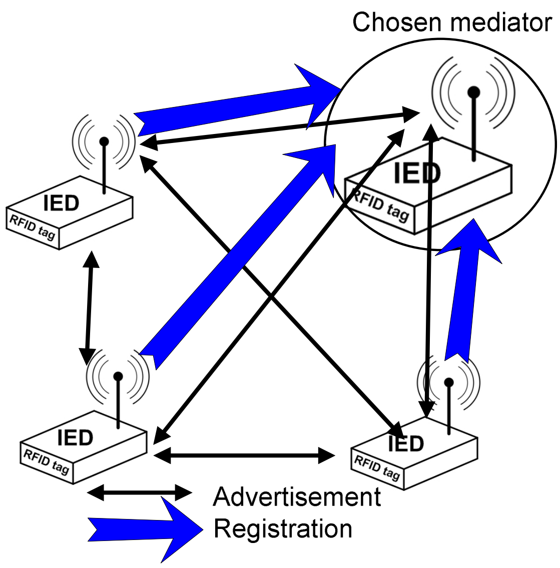

Fig. 4. The mediator selection process

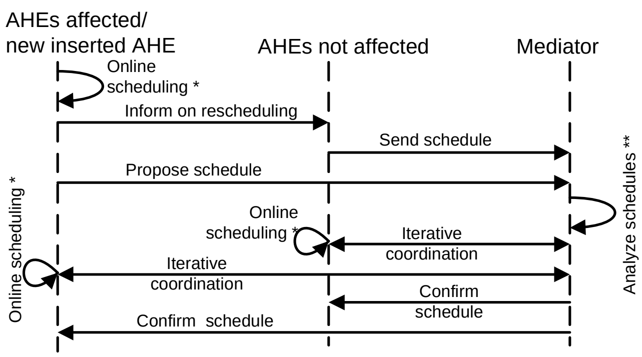

Fig. 5. Real-time resource allocation with a mediator agent

(*) **AHE on line scheduling algorithm**:

1. Invalidate schedule
2. default_operation_and_resource:=nothing
3. *For each possible processing operation* of the AHE that is not scheduled
   - 3.1. *For each resource* able do the selected operation
     - 3.1.1. If there are constraints from the moderator following an iterative coordination then *delay the time of current selection* (scheduled time range) by the requested time
     - 3.1.2. If the time of current selection is less then the time of default selection then *choose operation and resource*:=(operation selected at 3, resource selected at 3.1)
4. *Go to* 2 *and repeat* until all operations are scheduled

The new generated schedules are compared between them by the mediator: the AHE that did rescheduling informs all other AHE to *send their schedules* to the mediator. After *analyzing the proposed schedules*** and *an iterative coordination* mediator-agents in which resource overlapping (unfeasible schedules) are eliminated, the feasible ones are confirmed and production resumes.

(**) **Analyze schedules** (conflict resolution)

1. Form the GANTT resources chart using the proposed resource allocation from each AHE;
2. For each resource validate the proposed allocation starting with the AHE that arrives first; invalidate the proposed allocations that overlap and start a coordination dialogue with the associated AHEs.

Besides transportation and processing times, the resource allocation model contains: the status (operational/offline), the degree of occupation and the penalties inflicted to resources.

4. IMPLEMENTATION ASPECTS

The implementation of the proposed type of control was done in the pilot platforms of the two institutions (AIP and CIMR). According to the 2-layer control model described in Fig.1, real-time resource allocation was decoupled from long-term planning. This decoupling is supported by aggregating a *product* with a *pallet carrier* equipped with an *Intelligent Embedded Device*. The physical implementation of the AHE (Fig.6) consists of:

1. An Overo Air *processing module* (www.gumstix.net) based on an ARM processor, with WiFi communication, running Linux, configured for real-time applications;
2. A *transportation pallet with RW RFID tag*: this is the carrier of the product to be progressively manufactured, offering it transportation services;
3. *Product*: the part of the AHE being manufactured in a sequence of operations executed by assigned resources.

The infrastructure supporting the high level control consists of the PCs attached to the workstations and the cell server on which the global planning and scheduling application resides; this application can be relocated on any PC connected to the cell network infrastructure. The *product routing control* (low level) is done by a PLC which receives from each AHE standard files decoded to command the conveyor devices (motors, diverting units, and stoppers) so that the product visits the allocated resources and gets processing services.

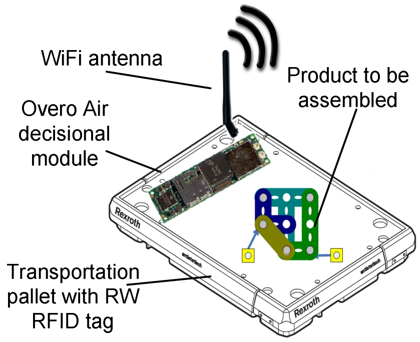

Fig. 6. Physical implementation of an AHE

Product localization is done by the PLC which reads the IDs of the pallets in fixed (e.g. conveyor diverting) places using a RFID system (*AHE Localization*), and offers this information to the exterior through a server. This information is then read by the AHEs which are continuously polling the PLC; when their own ID is detected by the PLC the location where the ID was read is associated with the corresponding pallet (*Raise event: Inform of localization* in Fig.7).

The localization events trigger a decisional process on the AHE which sends its decision to the PLC (*Request service*), this entity being in charge of its realization (*Perform service*). After completion of the requested operation, the PLC informs the AHE on the result (*Inform*) and why the result is negative.

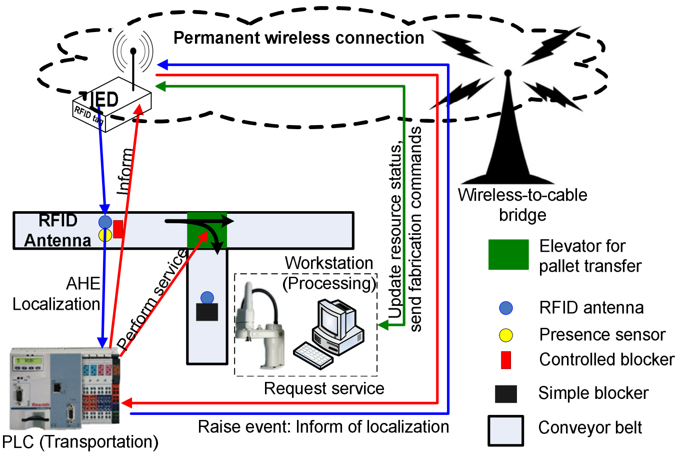

Fig. 7. Product localization and scheduling with AHE

The decisional software uses the Java Agent DEvelopment Framework. This framework is designed for developing MAS applications conforming to FIPA standards (www.fipa.org). It includes two main products: a FIPA-compliant distributed agent platform and a package to develop Java agents. The application provides the developer with an agent management system, a directory facilitator, an agent communication channel, debugging tools to aid developing multi agent applications, and intra-platform agent mobility.

Three important processes are implemented using the JADE application and its facilities:

- **Individual schedules**, computed from the perspective of each product, will be implemented as one-shot behaviours triggered by discrete events which perturb the normal operation;
- **Resource service access configuring**, done through a combination of a proprietary server application running on the resource controller and a middleware running on a PC and connected to a generic resource agent. This is necessary since resources may be heterogeneous, coming from different manufacturers, with different operating systems and interconnecting solutions (e.g.: Ethernet, serial port). This approach allows creating an intelligent infrastructure in which the resources are easily integrated and accessed using a common interface (a JADE agent);
- **Communication** between intelligent embedded systems and resources is done by utilizing common ontologies and interaction protocols easily defined in JADE, and is supported by the uniformity amongst agents, which are all implemented as JADE applications. Two types of agents were defined in the project: *product agents* in charge of the execution of the associated client order, which are active entities in the decisional process of resource allocation, and *resource agents* in charge of resource automation. These agents communicate in a mix of wire and wireless networks, allowing a high degree of mobility both from information and physical point of view.

5. CURRENT WORK AND CONCLUSIONS

The proposed 2-layer control model is currently tested in the pilot production platform of the Robotics & FMS Laboratory of the University Politehnica of Bucharest. The cell is composed of four material processing and quality control stations (robots, machine tools, vision), a pallet supply station (Cartesian robot) and a part supply station (dual part feeders with robot-vision management), linked to a close-loop multi-branch cell conveyor. Experiments were done in different scenarios; as an example, Fig. 8 shows a batch of 8 products of 4 distinct types which were planned, scheduled and executed in packets of 5 products. A resource breakdown was simulated during execution, (flash in Fig.8) causing resource rescheduling (see the two GANTT product charts).

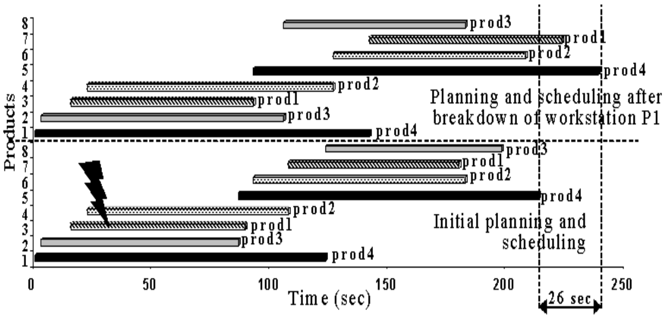

Fig. 8. Execution times before and after resource breakdown

The complete batch execution shows that the control system performs well even if affected by perturbations (only 26 sec, i.e. 10% increase of the total time, no interruption). Product prod3 on pallet 8 is executed faster after on line rescheduling, but makespan is greater because only 3 of 4 resources are available. Product rescheduling switches automatically to heterarchical mode, triggered by two types of events: station breakdown and missing parts. Experiments carried out on a batch production of 256 products put in evidence recovery times of 6.4 to 6.8 time units from resource failure to rescheduling of packet OH in execution [for dim (packet) =5] and resuming production, and of 83 to 136 time units from a local storage depletion to the generation of a Supply Holon, routing it to the empty storage, automatic storage re supply by the station robot and resuming production.

REFERENCES

Babiceanu, R.F., Chen, F.F., Sturges, R.H.: Framework for control of automated material-handling systems using holonic manufacturing approach, Int. J. Prod. Res., 42, 17, Taylor & Francis, 2004, p. 3551-3564

Barata, J.: The Cobasa Architecture as an Answer to Shop Floor Agility, Manufacturing The Future - Concepts, Technologies, Visions, Pro Literatur Verlag 2006, p. 31-76

Barták, R.: Mixing Planning and Scheduling to Model Complex Process Environments, PACLP2000, Manchester, UK, 2000

Borangiu, Th., Gilbert, P., Ivanescu, N.A., Rosu, A.: An implementing framework for holonic manufacturing control with multiple robot-vision stations, EAAI, 22, 4-5, Elsevier, June 2009, p. 505-521

Brussel, H.V. et al.: Reference architecture for holonic manufacturing systems: PROSA, Computers in Industry, 37, 3, 1998, p. 255–274

Cheng, F.-T., Chang, C.-F., Wu, S.-L.: Development of holonic manufacturing execution systems, Journal of Intelligent Manufacturing, **15**, 2004, Kluwer, p. 253-267

Leitao, P., Restivo, F.: ADACOR: A holonic architecture for agile and adaptive manufacturing control, Computers in Industry **57** (*2*), 2006, p.121–130.

Mayone, G., Naso, D.: A soft computing approach for task contracting in multi-agent manufacturing control, Computers in Industry, **52**, 2003, p. 199–219

McFarlane, D., Sarma, S, Chirn, J.L., Wong C.Y., Ashton K.: The intelligent product in manufacturing control, Journal of EAIA, July 2002

Meyer, G., Framling, K., Holmstrom, J.: Intelligent Products: A survey, Computers in Industry, doi:10.1016/j.compind, Elsevier, 2008

Murillo, J., Muñoz, V., Busquets, D., López, B.: Schedule coordination through egalitarian recurrent multi-unit combinatorial auctions, Applied Intelligence, 2009

Sauer O.: Automated engineering of manufacturing execution systems – a contribution to "adaptivity" in manufacturing companies, Proc. of the 5th Int. Conference on Digital Enterprise Technology Nantes, 2008

Trentesaux, D.: Les systèmes de pilotage hétérarchiques: innovations réelles ou modèles stériles?, Journal EU des Systèmes Automatisés, **41** (9-10), 2007, p. 1165-1202

Trentesaux, D.: Distributed control of production systems, EAAI, **22**, *7*, 2009, doi: 10.1016, Elsevier, p. 971-978