Multi-agent-based Agile Scheduling

R. J. Rabelo [1], L. M. Camarinha-Matos [2], H. Afsarmanesh [3]

[1] *Federal University of Santa Catarina, Department of Mechanical Engineering, Florianópolis (SC) Brazil, rabelo@gsigma-grucon.ufsc.br.*

[2] *New University of Lisbon, Faculty of Sciences and Technology, Monte da Caparica, Portugal, cam@uninova.pt*.

[3] *University of Amsterdam, Department of Computer Science, Amsterdam, The Netherlands, hamideh@wins.uva.nl.*

Abstract

The multi-agent systems paradigm represents one of the most promising approaches to the development of agile scheduling systems in manufacturing. Innovative and balanced perspectives of multi-agent approaches to agile scheduling are discussed in this paper, and several achieved results and current developments are described. The description of the multi-agent framework emphasizes the use of negotiation mechanism to improve the scheduling flexibility, as well as the aspects of information integration, communication and coordination in a community of agents. For information integration and management various approaches are addressed and the utilization of a distributed/federated database management system is suggested. Finally, some innovative aspects for agile scheduling are introduced, particularly in the context of virtual enterprise environments.

Keywords

Agile scheduling, multi-agent systems, integrated information management, virtual enterprise

1 INTRODUCTION

The application of a multi-agent system (MAS) approach to scheduling in manufacturing represents a very challenging test-case for the potential future system developments in this paradigm. Scheduling has already attracted various research efforts following a MAS approach, as exemplified by the following works [11], [22], and [19]. But, the richness of the scheduling application domain makes it the adequate framework, not only to assess the available techniques in MAS, but also to motivate new conceptual developments.

Scheduling in manufacturing, understood as the problem of suitable assignment of manufacturing resources to tasks / jobs within a specified time window and coping with a set of constraints, has been a major classical problem in manufacturing research and development. In the past the emphasis was given mostly to the scheduling optimality; later on however, the focus shifted to scheduling flexibility, and more recently the emphasis is directed towards the support for agility. This evolving scenario is due to the challenges provoked by the needs for global competitiveness in the open market, and the continuous evolution of both the manufacturing technologies and the socio-organizational criteria. An agile scheduling system provides the following two-fold capabilities:

- able to react dynamically in the presence of the events not previously foreseen in the current schedule, and

- consider the entire enterprise's production resources, beyond the traditional physical boundaries of the shop floors.

The first property characterizes the dynamic scheduling and leads to the need of a discussion of the "processing borders" and close integration of the production planning, scheduling, and execution supervision activities. The second property leads the enterprises to increase their business "flexibility boundaries" when supporting the virtual manufacturing and the virtual enterprise paradigms [13].

Trends in agile scheduling can be analyzed from several perspectives. This paper aims to discuss the innovative perspectives of some multi-agent approaches to advanced agile scheduling systems and represents an extension of the work first presented in [16]. It introduces some solutions and approaches to face one of the main general problems in the development of advanced scheduling systems, which is the fact that they require advanced information technology (IT) that itself changes progressively. Furthermore, the integration of human-based decision-making with scheduling functionalities is necessary. Therefore, the main challenge in innovative scheduling is to propose the balanced ways [14] to extend the life cycle of scheduling systems, so that it can support the new emerging production paradigms and the new socio-organizational structures in such a technologically dynamic scenario [20].

The research described in this paper is partially supported by the MASSYVE INCO-DC KIT Project, a cooperative initiative sponsored by the European Union, which involves Portugal (New University of Lisbon and the CSIN software-house), The Netherlands (University of Amsterdam), and Brazil (Federal University of Santa Catarina). The MASSYVE (*Multi-agent Agile manufacturing Scheduling Systems for Virtual Enterprises*) project aims to investigate the use of multi-agent systems in agile scheduling, and its extension towards the operation in a virtual enterprise environment. Taking a components-technology approach, the HOLOS framework [15] is adopted as a baseline for advanced scheduling. Some of the perspectives presented in this paper were already implemented in HOLOS, while some others are analyzed and implemented in the scope of the MASSYVE project. The information integration approach to support multi-agent systems in MASSYVE is based on another technology component, the PEER information management framework [1]. The ultimate goal of contributing to the scheduling problem in the context of virtual enterprises (VE) introduces a new and not so clearly defined dimension to the VE paradigm. As the coordination issues among members of a VE point to a high diversity of approaches [5], [6], a high flexibility - also at the scheduling level - is required.

2 THE HOLOS SCHEDULING SYSTEM

The HOLOS multi-agent scheduling system was initially designed to support individual manufacturing enterprises. After a brief introduction to the main features of this system, the proposed extensions towards a virtual enterprise will be discussed.

2.1 The HOLOS Multi-agent Framework

Creating a single general purpose scheduling system to cope with the requirements of every manufacturing system has proved to be a very difficult task. An alternative approach is to create a kind of development environment to support the derivation of scheduling systems customized to each particular operating context. HOLOS is a framework especially devoted to derive “instances” of agile scheduling systems [15].

A HOLOS scheduling system is represented by a group of agents configured for a particular shop floor and that process and exchange information about production orders in order to generate, execute, and supervise a schedule. In this framework an instance of a scheduling system is interactively and semi-automatically derived from a reference model – the HOLOS Generic Architecture (HOLOS-GA) [12] – supported by a specific methodology – the HOLOS Methodology [13], which uses the constructs introduced in Object Oriented Programming, and the power of the emergent approach of Agent Oriented Programming [21].

The derivation of an instance of HOLOS-GA for an enterprise is a very complex process that cannot be achieved by simply copying a derivation from one enterprise to another. In order to deal with this complexity, the HOLOS framework provides a semi-automatic / interactive system that "automates" most of the steps of the methodology – the HOLOS System Generator (HOLOS-SG) [13] – i.e., assists the human expert. The system derivation philosophy has been considered as a balanced trend for the development of complex systems [23]. In the case of HOLOS, it allows a system to be custom-tailored for each particular enterprise and, at the same time, to be reconfigured and adapted whenever new production methods, algorithms, production resources, etc., are introduced or changed.

2.2 Motivation for a Multi-agent Approach

The Multi-agent System (MAS) paradigm represents one of the most prominent approaches to build complex and flexible intelligent systems. The application of a MAS approach to agile scheduling is based on the idea that the scheduling agility can be extremely improved once it is based on the following key points:

i) distributed and autonomous systems instead of the centralized and non-autonomous solutions;
ii) negotiation-based decision making instead of the totally pre-planned processes;
iii) application of different problem-solvers in the same environment instead of only one fixed problem solver; and iv) concurrent execution instead of the sequential processing [24].

In summary, a multi-agent scheduling system is composed of a set of "processors" (nodes in a network of manufacturing resources), each one with its own particular capabilities (typically heterogeneous), that have to exchange and process information in order to contribute to finding a solution to the global scheduling problem. In spite of the lack of a common definition in the literature, a "processor" is considered as an *agent* when it possesses at least the following three properties [18]:

i) has a certain degree of autonomy to reason about and to make decisions by itself;
ii) has the capability to interact with other agents; and
iii) has the knowledge to independently solve a part of the global problem.

An agent can play several roles and behave in many different ways when it shows these elementary properties. One role it can play is to act *cooperatively*, that is the essential HOLOS agents' behavior, instead of acting destructively or selfishly. Cooperative scheduling ascends in significance in the complex manufacturing environments, while it is usually highly constrained. This means that, in many cases, a feasible and robust multi-agent scheduling can only be generated and executed with the dynamic, flexible and intelligent relaxation of the constraints within the distributed agents, i.e. with real cooperation. The more efficient this cooperation process is, the more efficient the agile reaction of the entire production structure will be, and hence a better information quality is provided to support a rapid decision-making. However, when the manufacturing process is extended towards a VE environment, this cooperative assumption cannot be necessarily guaranteed.

2.3 Classes of HOLOS agents

A HOLOS system is constituted by a set of instances of the following HOLOS agents' classes, the basis of the HOLOS-GA:

- *Scheduling Supervisor (SS)*: instance agent that performs the global scheduling supervision and it is the unique system's "door" to other systems.
- *Enterprise Activity Agents (EAA)*: instances that are directly associated to the manufacturing resources (robots, CNC machines, etc.), representing them into the agents community. These agents are the real executors of manufacturing orders.
- *Local Distribution Centers (LDC)*: instances that represent functional clusters of EAAs in order to avoid announcement broadcasting. They are also responsible to select the most suitable agent for a certain order after a negotiation process.
- *Consortium (C)*: temporary instances created to supervise the execution of a given order by the involved EAAs.

Figure 1 illustrates a general scenario of a particular HOLOS scheduling system. The scheduling system is viewed as an application that receives / feeds data from / to a CIM Information System (CIM-IS), receiving / providing information from / to other applications. Unlikely the classical systems, HOLOS is not a unique and comprehensive system, but rather a collection of distributed processes with some autonomy, independence and communication capabilities (agents).

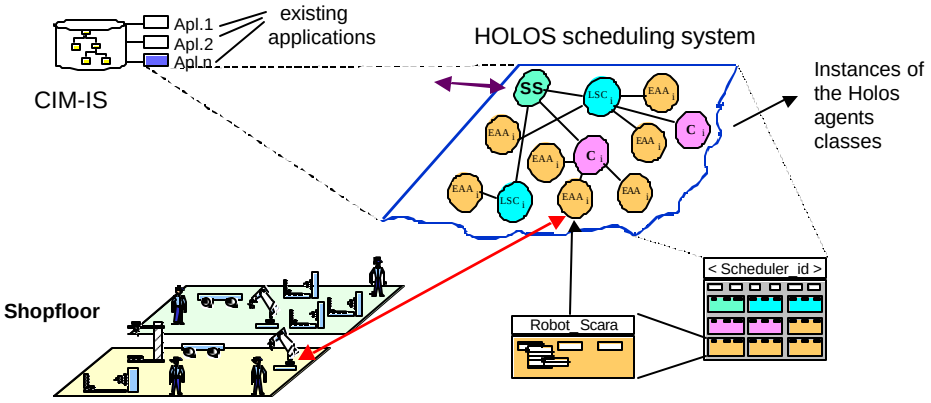

Figure 1 Example of a general MAS scenario.

Once launched in the computational environment, all the instances created by HOLOS-SG become persistent, except the Consortium agents. One Consortium is alive and active as long as the task it supervises is not completed. Once the task is finished the corresponding Consortium dismantles itself after generating some log information for auditing purposes. In HOLOS, there is not a unique global and comprehensive schedule, but rather a collection of distributed and inter-related pieces of smaller schedules. The Consortium concept allows an enterprise to improve many aspects of manufacturing flexibility, including the internal routing, production, and shop floor organization, providing a basis to overcome some limitations of the Group Technology concept as well as to support the virtual manufacturing paradigm. A consequence of this distributed approach is the difficulty in guaranteeing that the global schedule is optimal.

The EAAs are the tasks executors, linked to the "real manufacturing entities", that "sell" the services that can be provided by the production resources they represent. The other agents are designed only for control and co-ordination purposes. The essential information control flow used in the negotiation-based scheduling is illustrated in the Figure 2. The CIM-IS represents

a CIM Information System (see chapter 3). Therefore, a schedule is generated and supervised via a cooperative and tightly coordinated information exchange among agents.

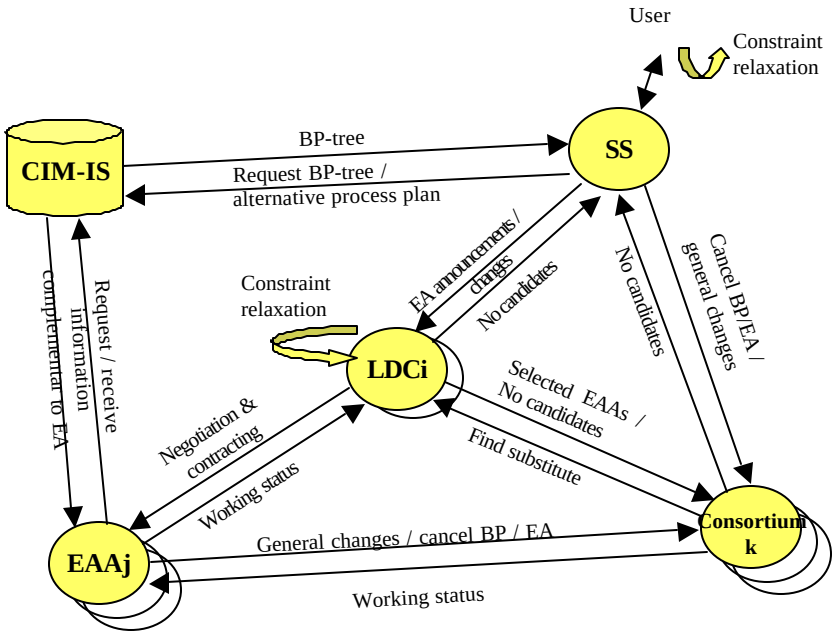

Figure 2 – Interactions between HOLOS agents.

2.4 Negotiation in Scheduling

A major difficulty with classical scheduling systems is handling of conflicts. Several problems can arise during the schedule generation, after its generation, and during its execution, such as the temporal, capacity, or technologic *conflicts.* These problems may come from the planning, scheduling or execution supervision activities. There are several methods that can be applied for the conflict resolution in a multi-agent system. HOLOS uses the Contract-Net Protocol coordination mechanism to support the task assignments to agents, and the Negotiation [7], [17] method to overcome conflicts taking place during one of the three mentioned scheduling phases.

Figure 3 illustrates how the negotiation approach is used in HOLOS. Notice that the main function of a scheduling system is to assign tasks (production orders) to manufacturing resources (robots, CNC machines, workers, etc.), represented by agents, during certain periods of time. The basic procedure is to:

(a) announce a task (an "enterprise activity", which is modeled as an object and represents the requirements of an individual process plan operation) through the MAS network and then make the agents exchange information about it with other agents, and so that
(b) an agent is selected to perform such task at the end of this process [11].

The coordination of this entire network-based negotiation process comprises various phases and requires a specific high-level protocol. The developed HOLOS Negotiation Protocol supports the information exchange associated with each of these phases.

The structure of the multi-agent control hierarchy has a high impact on the system performance. The effects of a system's topology can be evaluated through measuring several types of costs, such as the execution (normally expressed in units of time spent in the execution of a task), coordination, and vulnerability costs. In HOLOS the interaction between the agents is only vertical and agents cannot change the set of other agents that they can communicate with. Considering the classification schema proposed in [9], the HOLOS control hierarchy is *functional with small processors*, using one *global manager* (the agent

SS), some *functional managers* (the agents LDC), and assuming that a shop floor is usually composed by production resources (the agents EAA) with small production capacities.

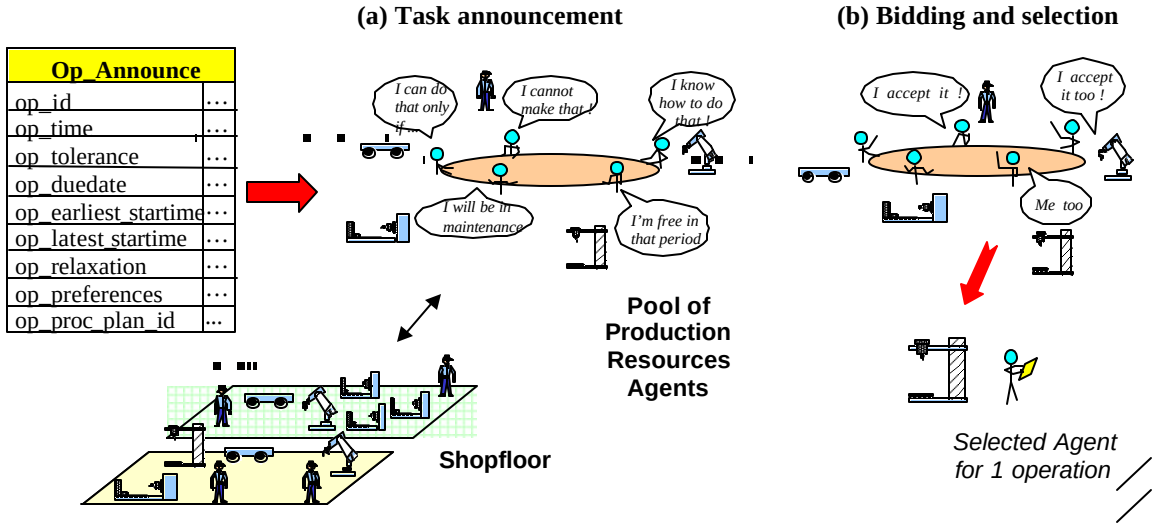

Figure 3 – Negotiation in scheduling.

Inter-agents communication is a crucial aspect in MAS-based solutions. When it comes to implementation, this is a difficult aspect due to the strong heterogeneity in industrial environments. A HOLOS agent can establish communications with four kinds of external entities: other HOLOS agents, the end-user (giving the possibility to both supervise the system and to intervene in some situations), a CIM Information System and the production resources. Resorting international or *de facto* standards for these communication processes, the information exchange becomes easier and faster. Another advantage that can be mentioned here is the increase of the system's life cycle, as there is no need to constant changes in the protocols when new heterogeneous systems are added to the enterprise. Within HOLOS, the MAP/MMS [10] is used as the supporting high-level protocol for communication with the production resources, and the STEP/SDAI [8] is used to communicate with the CIM Information System (CIM-IS). The HOLOS Negotiation protocol which is of a higher level than KQML, is used to support the high-level communication between agents. Figure 4 illustrated this complex and heterogeneous scenario. For example, *"mm_LDContent (<machine_id>"* is the MMS statement sent by a certain EAA to the production resource *machine_id* to start the execution of a program stored in its memory. In the case of SDAI, the statement *"is_get_instance (process_plan, <pplan_id>)"* for instance, is used to access a certain *process plan* called *<pplan_id>* in the CIM-IS. The statement *"send_msg (<Consortium_id>, <SS_id>, < (OCS, <task_id>, <status>))"* would be the answer of the agent *<Consortium_id>* to the *<SS_id>* regarding an earlier query about the status of a certain task. This message is semantically classified as *"OCS"* (Operational Control Status) in the Negotiation Protocol. Having different protocols that simultaneously run in the same environment, requires the interoperability among protocols.

Another important and very demanding task is the integration of the scheduling system with the real production resources, since the controllers of many manufacturing devices are not *open* enough. In some cases it might even imply retrofitting some machines, e.g. due to the introduction of new sensors, new I/O functionalities, etc. This integration is necessary in order to make the EAA agents "talk" directly to the manufacturing resources (according to the MMS protocol, in this case) to send commands and to receive their status, for instance in real-time. A balanced approach for the migration from existing components towards an integrated manufacturing infrastructure can be achieved via the "agentification" (*wrapping*) of

manufacturing resources [3]. In other words, this agentification aims at transforming the manufacturing resources into agents that can cooperate in a multi-agent community (Figure 4).

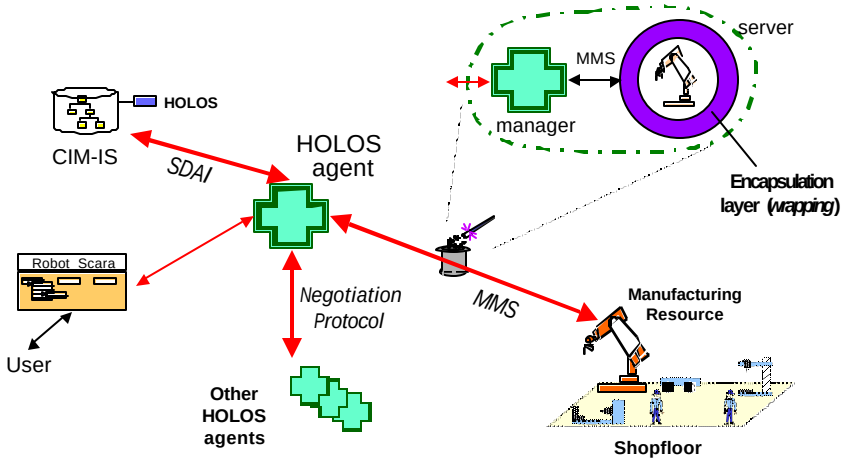

Figure 4 - The integration of existing controllers in an object-oriented framework.

This agentification process usually involves the following steps:

i) building an adapting layer around the existing controllers in order to transform them into normalized servers; in the case of human resources, a human front end is necessary; and

ii) building an agent manager, which includes a high level "mirror" of the resource's functionalities.

As it can be noted, an EAA agent is, in fact, a "transparent" tandem aggregation of two processes (Figure 4): the *server* itself (as a wrapped resource controller) which executes the tasks, and the *manager* which manages the server's agenda and "sells" the server's capabilities to the agents community. The main advantage of this tandem architecture is its efficiency, due to the resulted parallelism that allows a speed up of the entire scheduling process. While the manager continues to negotiate the server's capabilities, the server keeps executing its contracted tasks.

In terms of implementation, a HOLOS agent is a software module (computational process) modeled in frames, and having the following basic components:

- the manager (including a *knowledge base*, which contains the agent's functionalities), an *agenda* (which describes the list of scheduled tasks);
- a *mailbox* (which describes the tasks under negotiation);
- the *communication protocol interfaces;* and
- in the case of the EAA agents, the description (*mirror*) of their capabilities.

Figure 5 shows the detailed architecture of a HOLOS agent. In the first prototype version, the agents were implemented in Prolog, extended with a frame layer, and the communication services were written in C. The HOLOS scheduling system / agents run on LINUX / PC machines [1].

The scheduling system is viewed as an application that receives / feeds data from / to a CIM Information System. Differently from classical systems, HOLOS is not a unique and comprehensive system, but rather a collection of distributed processes (agents). Each agent has its graphical user interface. In the EAA agents case, they are linked to the production

[1] The system is currently being re-implemented on a PC/Windows-NT/C++ platform.

resources they are associated to. For example, a certain EAA, identified as a "Robot_Scara", can establish a communication with the corresponding physical entity existing in the shop floor.

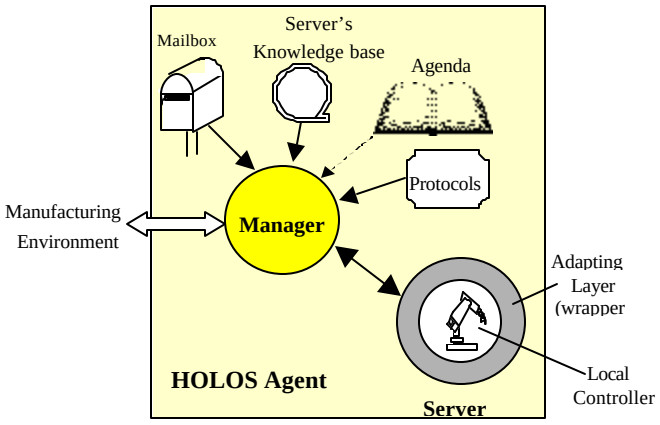

Figure 5 – The architecture of a HOLOS agent.

3 INFORMATION MANAGEMENT FOR MASSYVE SCHEDULING

As it is mentioned in section 2.3, the architecture of the HOLOS system includes the CIM-IS component for the integration and management of information among its agents. Through the CIM-IS, different systems in the enterprise can share and access all the information they need, in an integrated and transparent way, that is vital for the agile decision-making (see Figures 2 and 7). The CIM-IS can be regarded as the *logical aggregation* of the information structures (also called an "Integrated Schema" in a *federated database* architecture). In other words, the physical implementation of the CIM-IS can be achieved in several ways, e.g. with centralized or distributed/federated databases. In this section first the data and functional requirements of HOLOS scheduling is addressed. Then three approaches to the implementation of the MASSYVE information management is described. The PEER federated database system is also briefly described in section 3.3, since it is used in two of the implementation approaches considered for MASSYVE.

3.1 Variety of information sources for scheduling

Agile scheduling requires access to a wide variety of information items during both its generation and execution (see Figure 6). Some of this information is "indirectly" provided through the MRP activity of the Production Planning, while other information is collected directly from several other sources. At the same time, the information about the production plan, the process plan, the product model, and the production resources models need to be gathered from several sources, in order to be used by the scheduling system.

3.2 Agent functionalities and their required data

In the HOLOS architecture, every kind of agent plays a certain role, for which it requires to also access some data either from other agents, or from other sources of information. Some of the agents' functionalities are described and their required data is identified:

- A scheduling starts when the SS agent consults the CIM-IS to check if there is a production plan with the list of tasks that needs to be scheduled. During the scheduling execution, the process plan models need to be accessed.

- The SS agent is the only agent that is visible outside HOLOS. Any other system, tool, or service, which needs to communicate with the scheduling system and access any scheduling related data, must contact the SS agent.
- Besides the SS agent, the only other kind of agent that has direct access to the CIM-IS is the EAA. The EAA agents access the CIM-IS during the scheduling generation, in order to get information about the tools, NC program, etc., necessary to perform their tasks.
- Once the scheduling system is derived, the production resources model is not needed anymore for the scheduling purposes. But the execution status of the production resources is needed to be accessed by the EAAs. Also the "scheduling models" (containing both the general information about the schedule of the tasks and the general status of the EAAs) must be made available in the CIM-IS, in order to be accessed by other activities / tools. The capability status is also needed to be stored in the CIM-IS, since they can change during the life time of the EAAs.
- Several other activities, for instance the planning, need to have up-to-date information about the schedules in order to modify the enterprise's production strategy and to guide the enterprise's logistics plan, that is fundamental in a global and competitive economy.

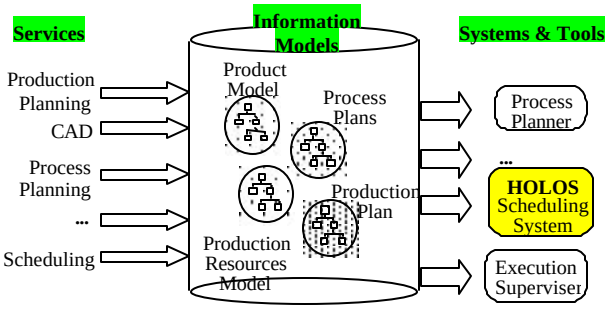

Figure 6 – The central CIM-IS.

3.3 PEER Federated database system

The PEER federated database system, developed at the University of Amsterdam, is an object-oriented information management system. It supports the sharing and exchange of information among cooperating autonomous and heterogeneous agents (or nodes) without the centralization and/or data-redundancy. In PEER, interdependencies between two nodes' information are established through the conceptual schemas defined on their information; thus there is no need to store the data redundantly in different nodes. Every node is represented by several schemas:

- a local schema (LOC): the local schema is the schema that models the data stored locally.
- several import schemas (IMP): the various import schemas model the information that is accessible from other databases.
- several export schemas (EXP) : an export schema models some information that a database wishes to make accessible to other nodes (usually, a node defines several export schemas)
- an integrated schema (INT): the integrated schema presents a coherent view on all accessible local and remote information. The integrated schema can define a particular classification of objects which are classified differently by the schemas in other nodes.

A prototype of the PEER system has been developed in the C language for the UNIX environment, together with an interface that allows PCs to also have access to it.

3.4 Different Approaches to information integration in MASSYVE

The agile scheduling system, as described in HOLOS, requires an advanced database management system that supports all the requirements described in sections 3.1 and 3.2 above. Depending on the configuration of the environment, different approaches may prove to be more appropriate. In this section, three implementation architectures are presented and some of their characteristics are briefly discussed (figure 7).

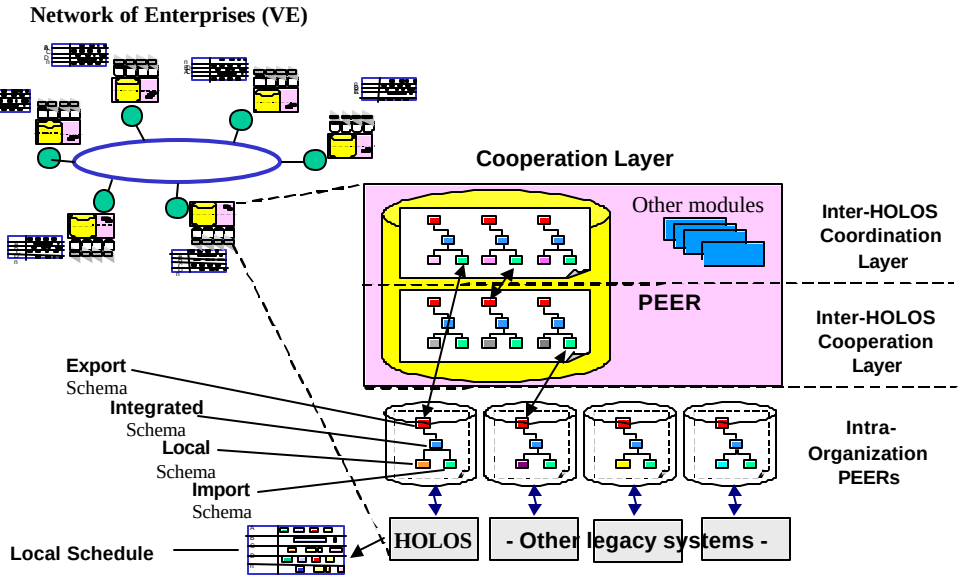

Figure 7 – PEER-based implementation architecture for MASSYVE.

In the existing implementation of the HOLOS system, a centralized implementation strategy is utilized, that sufficiently and efficiently supports the information access from different systems inside one enterprise. However, as with any centralized database, and depending on the size and complexity of the enterprise, there are certain disadvantages associated to this approach that can be better resolved for MASSYVE, if considering a distributed / federated architecture. A main problem is the vulnerability of the CIM-IS system, due to the centralization of all data in one node. A bigger problem will also rise in the case of a virtual enterprise, where a group of enterprises are involved. In such an environment, the centralization becomes a real bottleneck, both in the technical and organizational sense (see next section). Another problem is the inefficiency, which will rise in the case of a big enterprise that constitutes several production sites. In this case, the CIM-IS needs to manage the integration of large amounts of updated data from several physically distributed sources, as well as the large number of information access requests from those systems (see the last section).

On the other hand, it is also not convenient to model the current centralized implementation of the HOLOS system using PEER, because PEER is also essentially devoted to integrate distributed applications. So, it is not a good solution if we use PEER as a centralized database, through transforming the CIM-IS into a PEER node. Thus, considering the future of manufacturing systems and enterprises, and in order to satisfy both the case of virtual enterprises, and the case of big multi-site manufacturers, MASSYVE proposes a multi-layered federated database architecture. This architecture supports the sharing and exchange of information both within each multi-site enterprise and among different enterprises uniformly.

A three-layered federated approach, based on PEER is described below as one implementation design.

Intra-organization Federated Layer. Clearly, the federated database architecture can also be applied within each enterprise [2]. If the components of an enterprise are distributed and run on different systems, the communication and exchange of information among different agents in HOLOS can benefit from the federated approach. This approach is illustrated as the "intra-organization PEERs" in figure 8, representing a decentralized CIM-IS for HOLOS. The HOLOS scheduling system constitutes then one PEER node, and the other applications (production planning, process planning, etc.) are also individually represented by other PEER nodes. Each system application in turn can comprise other PEER nodes if necessary. Namely every agent, such as an EAA, can store and manage its generated information in its internal PEER database. Furthermore, different agents can share and exchange information via PEER to PEER access. In this architecture no centralization of information is required.

Federation of HOLOS systems. Agile scheduling under a more advanced perspective should also be able to support *virtual manufacturing*, i.e. to have the capability of considering the manufacturing resources of other enterprises, in a tight cooperative way of operation. This approach is illustrated as the "Inter-HOLOS Cooperation Layer" in figure 8. In this case, PEER is logically considered as a *cooperation layer* for information exchange supporting a set of software modules that act as an intelligent front-end between an enterprise and its partners (for a given business). This approach is similar to the one developed in the PRODNET II project [4]. Through the federated database schemas, PEER can offer an integrated and reliable way for information exchange regarding the manufacturing resources of the other enterprises that one enterprise "manages".

Federation of Virtual enterprises. This approach corresponds to an upper layer on the previous one, and it is illustrated by the "Inter-HOLOS Coordination Layer" in figure 8. A virtual enterprise is a group of enterprises representing physically distributed sources of information. These enterprises need to cooperate and exchange information in order to jointly fulfil certain specific business opportunity. While intra-enterprise scheduling can be supported even by the centralized approach, the centralized architecture is insufficient for the inter-enterprises scheduling. This environment is by nature "distributed" and consists of a given set of autonomous nodes. PEER can support this complex domain, by considering every enterprise as a node in the federation, which maintains its local autonomy on the data and defines a set of export schemas through which the data is made available to other specific nodes [2]. Furthermore, every node will be able to import data from other nodes through their import schemas, and have access to their data according to the pre-defined access permissions. As a consequence of this general interaction facility, the approach allows not only the cooperation but also supports the coordination of the federated nodes (the group of enterprises involved in the distributed schedule) in order to accomplish a common or global task, while the local autonomy and independence of every node is preserved and reinforced.

The multi-layered federated database approach properly supports the evolution of the scheduling system through its life-cycle, with no centralized repository of data or control and no need for data redundancy in the network.

4 EMERGING MULTI-AGENT PERSPECTIVES TO SCHEDULING

As previously explained, the MASSYVE architecture is based on the above described HOLOS and PEER systems. This chapter introduces further promising extensions to the HOLOS architecture in terms of MAS approaches to negotiation-based scheduling (Figure 8). The extensions being evaluated in MASSYVE are divided according to three main perspectives: new organizational structures of multi-agent systems, scheduling in a VE environment, and contributions to workflow and project management.

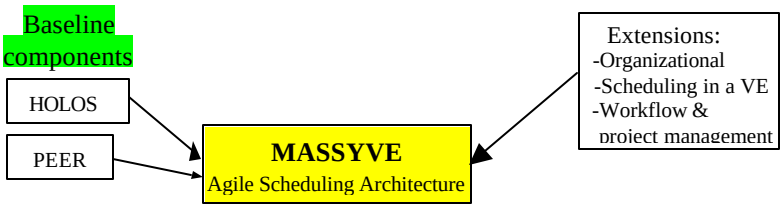

Figure 8 – Contributions to MASSYVE architecture.

4.1 New MAS organization structures

The scheduling system can be improved by increasing the level of flexibility in the agents concerning their behavior in the hierarchical control structure. As mentioned in the section 2.3, the HOLOS structure is fixed in terms of types of interactions, agents' knowledge and negotiation capabilities. However, several extensions can be added to the system. These extensions can improve not only the quality of a schedule, but also the quality of the strategic, tactile and operational production plan. Examples of such extensions identified in MASSYVE are:

- *Support for dynamic assumption of roles by agents.* The agents' pre-defined roles (i.e. their functionalities and levels of delegation) in a system can be dynamically changed or extended to deal with different tasks or situations, both in a virtual enterprise (VE) scenario and in an individual (local) enterprise.
- *Define criteria for selection of agents and their relation to the global optimization criteria.* In HOLOS, the best bid from the (EAA) agents is selected by the LDC agents based on fixed criteria, using the classical contract-net philosophy. The levels of interaction / cooperation between these two classes of agents could be improved if the bidders could "iteratively" send to the LDCs their preliminary proposals (bids), asking if they are satisfactory or not, instead of sending final proposals at once. This can decrease the EAAs' local ignorance and can improve the quality of the LDCs' selection process. Another approach is to consider a stronger human intervention in this process, i.e., a less automatic process.
- *"What-if" supported negotiation.* Like in MRP-II systems, the scheduling system could allow a schedule generation to be *simulated* (and hence evaluated) specially for capacity and logistic planning purposes. This means that EAAs would have to reason in two alternative "spaces": the real one or the simulated scenario. Eventually multiple scenarios might be kept during the what-if session.
- *Order splitting negotiation.* In a classical scheduling scenario the splitting of orders is decided by the production planning activity. A different approach is to allow the EAAs (the scheduling activity) to offer splitting proposals based on their local agendas (or even based on their skills). This leads to an increase in the level of the agents' autonomy. This capability can be useful for a better resource management as a way to avoid bottlenecks, to decrease the risks caused by a machine failure or to improve flexibility in failure

recovery, and to find EAAs when there is no one able to accomplish a full order, leading to a more balanced workload distribution.

- *Design of the organizational structure according to the needs of the application area.* Other scheduling application areas, namely in the case of virtual enterprises, may require other more efficient configurations of the multi-agent system architecture and of the properties of their agents.
- *Bargain in negotiation.* Using a more sophisticated negotiation protocol, the EAAs can *bargain* with the LDCs about a certain order so that their agendas (i.e. the machine utilization) can be maximized. This also improves the agents' autonomy and the decentralized decision-making capability.
- *Access to agents' internal status.* Acting in a pro-active way, the high level supervisor agents (the Consortium agents, for instance) could be permanently checking the status of their "slaves", even for those having reporting capabilities (because sometimes the agents can lie or become non operational). This feature can be applied both in a VE scenario and in an individual enterprise.
- *Organization for shared resources.* EAAs (production resources) can work for several enterprises "simultaneously". In spite of a given production resource belonging to an enterprise, it could be supporting another one, like a resource (partial) rental.
- *Forecasting in Scheduling.* Utilization of forecast information by the agents (mainly the EAAs) in order to improve their local schedules. In this sense, the agents can make bids also based on forecasting (seasonal products, pre-defined clients orders, weather conditions, etc.).

4.2 Scheduling in a VE environment

Another set of extensions to the scheduling system more specifically oriented to cope with the VE environment were also identified by MASSYVE. Many of these facets can be also applied to the case of workflow and project management.

- *Coping with agents that are not totally cooperative.* Although a general willingness to cooperate can be assumed when enterprises join their manufacturing efforts in a VE, variable levels of trust have to be considered. This means that the agents representing different enterprises may be not cooperative, but rather *selfish* (they are not interested in a global optimization but only in themselves), or even *destructive* (competing enterprises, for instance, which send wrong information on purpose). The cooperation can be just one strategic behavior that a selfish enterprise, for example, can exhibit. Therefore, the agents may lie or even hide information, which changes the general assumptions followed in HOLOS and many other MAS systems.
- *Negotiation with incomplete or imprecise information.* The classical negotiation protocols require, from the agents, a very well established sequence of complete information exchanges about a task in order to execute it. There are, however, several application domains, in which when an order is initially received it may show incomplete or imprecise data, being progressively complemented in a later phase [4]. It is therefore necessary not only to adapt the agents and the protocols to allow reasoning with partial information, but also to prepare the production systems accordingly.
- *Global-contract-biased negotiation.* MAS-based solutions have the local ignorance as an intrinsic problem. Local ignorance can lead an agent to disregard useful information in its reasoning process that is related to a global negotiation contract (and not to individual orders). Therefore, more efficient knowledge propagation strategies have to be applied in order to provide all the involved agents with the necessary information. This aspect has to be considered in conjunction with the coordination policy established for each particular VE, as well as the corresponding information access rights and visibility.

- *Pre-defined or dynamic clusters of agents.* The agents can dynamically change their relationship to some nodes instead of being associated to fixed groups of agents, including the agents of other enterprises. These relationships should be determined by the *contractual* rules between VE members. Definition and management of *contracts* is therefore an important item associated to the MAS organization.
- *Authority, accessibility to "internal" agents of a node, and organizational structure.* In HOLOS, for instance, all interactions are vertical, i.e. agents of the same class (the same hierarchical level) cannot communicate with each other. However, in a wider scenario (like in a VE one) in which a global resource management is desired, horizontal interactions may be necessary.
- *Volatility of agent skills.* The agents' capabilities can change along their life cycle (new, change and/or loose of capabilities), which requires from the system as a whole a high degree of "self-adaptation". Although this could happen with a single EAA (a machine could loose some of its functions), the situation is more natural in the context of a VE.
- *Benefits of a VE schedule vs. individual schedules.* A VE scenario requires a more cooperative relationship between its members than the one in traditional business relations between enterprises. So it makes sense to speak of a global VE schedule (global profits) instead of a collection of individual and independent schedules.
- *Competing scheduling.* In a VE scenario, each enterprise normally has its own (local) schedule. However, depending on the task to be performed, each of the local schedules may "compete" with each other without considering a global gain. New strategies are necessary to handle this structure.
- *Distribution logistics scheduling in electronic commerce.* Electronic Commerce is becoming more and more important. One of the bottlenecks for its implementation in some sectors is the distribution logistics. More specialized scheduling functionalities are required for this area.
- *Perturbation analysis (changes in behavior).* In a VE, the supply-chain has to be constantly monitored in order to guarantee realistic global schedules and to keep an updated logistic plan. In this sense, the agents / systems have to be sensitive to problems and to react accordingly.
- *Negotiation in VE using mobile-agents.* Mobile agents can be given a "negotiation mission" to be accomplished in another enterprise. Thus, instead of making the two enterprises negotiate via direct interactions, a mobile-agent can be moved to an enterprise to perform most of the automatic negotiation tasks locally. Once the process is finished, it can be moved again to another enterprise of the VE to perform other negotiation steps, even taking the previous knowledge into account. This approach has the advantage of not depending on the permanent availability of network connections between the enterprises.

4.3 Workflow and Project Management

The extended facets described in previous sections can be generalized to other areas. For instance, its application seems particularly promising in the areas of workflow and project management systems.

In spite of the recent progresses in the workflow management systems, they are not very flexible in terms of dynamic changes of the workflow plans. Similarly, the area of project management support tools, specially when involving consortia formed by people from different organizations, can benefit from flexible negotiation-based scheduling and supervision. In particular, the introduction of dynamic scheduling, relaxation of constraints, tasks re-assignment, etc., are particularly promising here. Humans can be considered as a special kind of agents that need to be taken into account in a workflow execution or project management. When interacting with these agents, however, some of the basic assumptions of

HOLOS, namely the assumption that agents are benign / totally cooperative, do not hold. Human agents may hide information or even lie for their own benefit. Figure 9 illustrates a scenario to support the decision on how to manufacture a given part, if in the shop 1 or in the shop 2, also regarding the possibility for a human intervention / decision.

The generalization of the described approaches to these areas is another aspect being addressed by MASSYVE.

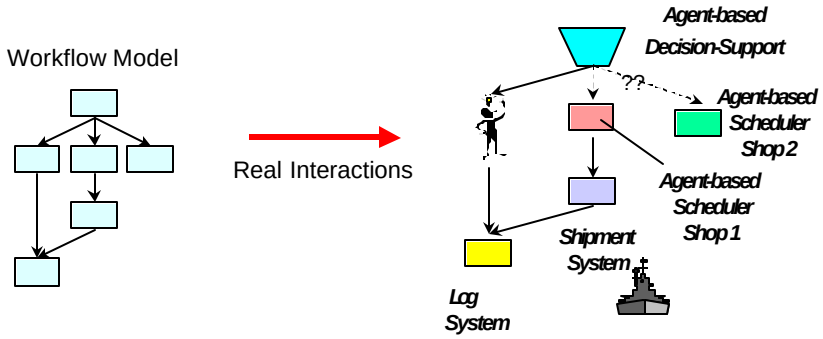

Fig. 9 – Multi-agent negotiation and execution in dynamic workflow

5 CONCLUSIONS

Multi-agent systems and the negotiation-based task assignment represent an effective approach to the development of agile scheduling systems. An initial implemented prototype system has shown the feasibility of the approach in the area of shop floor scheduling. In this paper, a number of possible extensions to the implemented architecture were identified in order to increase its agility and to support scheduling in a virtual enterprise environment.

One of the improvements discussed is the integration of the multi-agent architecture with a federated database management approach. This new research direction seems particularly suited to the requirements of a VE. One challenge to previous approaches is the handling of agents that are not necessarily totally cooperative, as is the case in a virtual enterprise. Another key issue is the scheduling in the context of imprecise and incomplete task information, as a result of the management of incomplete and imprecise orders in a supply chain.

These aspects represent some of the issues being researched in the framework of the MASSYVE project.

ACKNOWLEDGEMENTS

This work is funded in part by the European Commission, project INCO-DC KIT MASSYVE number 962219, and by the CNPq (The Brazilian Council for Research) under the grant 380009/97-1.

6 REFERENCES

[1] Afsarmanesh, H.; Wiedijk, M.; Hertzberger, L. (1994) Flexible and Dynamic Integration of Multiple Information Bases, Proceedings DEXA'94 - 5th IEEE International Conference on Databases and Expert Systems Applications, Springer-Verlag, pp. 277-288.

[2] Afsarmanesh, H; Garita, C; Hertzberger, L.O.; Santos, V. (1997) Management of Distributed Information in Virtual Enterprises: The PRODNET Approach - in proceedings of the 4th International Conference on Concurrent Enterprising (ICE'97).

[3] Camarinha-Matos, L. M.; Rabelo, R.J.; Osório, L. (1996) Balanced Automation, in Management and Control of Manufacturing Systems, ed. S. Tzafestas, Springer-Verlag, pp. 376-413.

[4] Camarinha-Matos, L. M.; C. Lima; Osório, L. (1997) The Prodnet platform for production planning and management in virtual enterprises, Proceedings ICE'97 - 4th International Conference on Concurrent Enterprising, pp. 385-406, Nottingham UK.

[5] Camarinha-Matos, L.M.; Afsarmanesh, H. (1998) Flexible coordination in Virtual Enterprises, Proceedings of IMS'98, 5th IFAC Workshop on Intelligent Manufacturing Systems, Gramado, Brazil, 9-11 Nov 98.

[6] Camarinha-Matos, L.M.; Lima, C. (1998) Configuration and Coordination issues in a Virtual Enterprise Environment, Proceedings of PROLAMAT'98 – IFIP Int. Conf. On Globalization of Manufacturing in the digital communications era of the 21st century, G. Jacucci, G. J. Olling, K. Preiss, M. Wozny (Eds.), Kluwer Academic Press, ISBN 0-412-83540-1, Sep 98.

[7] Davis, R.; Smith, R. – Negotiation as a Metaphor for Distributed Problem Solving, Artificial Intelligence, N 20, pp. 63-109, 1983.

[8] Fowler, J. (1992) Proposal for the STEP Data Access Interface Specification, STEP Implementation Specifications Committee, NIST.

[9] Malone, T.; Smith, S. (1988) Modeling the Performance of Organizational Structures, International Journal on Operations Research, Vol 36 N 3, pp. 421-437.

[10] Mackiewicz, R. (1994) An Overview on the Manufacturing Message Specification, in http://litwww.epfl.ch/~mms.

[11] Rabelo, R.J.; Camarinha-Matos, L. M. (1994) Negotiation in Multi-agent Based Dynamic Scheduling, Journal on Robotics and Computer Integrated Manufacturing, Vol 11 N 4, pp. 303-310, Pergamon.

[12] Rabelo, R.J.; Camarinha-Matos, L. M. (1995) A Holistic Control Architecture Infrastructure for Dynamic Scheduling, in Artificial Intelligence in Reactive Scheduling, Eds. Roger Kerr e Elizabeth Szelke, pp.78-94, Chapman & Hall.

[13] Rabelo, R.J.; Camarinha-Matos, L. M. (1996a) Deriving Particular Agile Scheduling Systems using the HOLOS Methodology, International Journal in Informatics and Control, Vol 5 N 2, pp. 89-106, Romania, June.

[14] Rabelo, R.J.; Camarinha-Matos, L. M. (1996b) HOLOS: a methodology for deriving scheduling systems, in Balanced Automation Systems - Architectures and Design Methods, Eds. Luis M. Camarinha-Matos and Hamideh Afsarmanesh, Chapman & Hall, pp. 181-194.

[15] Rabelo, R.J. (1997) A Framework for the Development of Manufacturing Agile Scheduling Systems – A Multi-agent Approach, Ph.D. Thesis, New University of Lisbon, Portugal.

[16] Rabelo, R.J.; Camarinha-Matos, L. M., Afsarmanesh, H. (1998a) Multiagent Perspectives to Agile Scheduling, in Intelligent Systems for Manufacturing – Multi-Agent Systems and Virtual Organizations, Eds. L. M. Camarinha-Matos, H. Afsarmanesh and V. Marik, Kluwer Academic Publishers, pp. 51-66.

[17] Rabelo, R.J.; Camarinha-Matos, L. M. (1998b) Generic framework for conflict resolution in negotiation-based agile scheduling systems, Proceedings IMS'98 – 5th IFAC Workshop on Intelligent Manufacturing Systems, Gramado – Brazil, pp.187-192.

[18] Sichman, J.; Demazeau, Y. (1992) When can knowledge-based systems be called agents ?, Proceedings IX Brazilian Symposium on Artificial Intelligence.

[19] Schmotzer, M.; Paralic, J.; Csonto, J. (1998) Scheduling in a Multi-Agent Environment, in Intelligent Systems for Manufacturing, L.M. Camarinha-Matos, H. Afsarmanesh, V. Marik (Eds.), Kluwer Academic Publishers, ISBN 0-412-84670-5, Aug 98.

[20] Smith, S.; Lassila, O. (1994) Configurable Systems for Reactive Production Management, IFIP Transactions (B-15) on Knowledge-Based Reactive Scheduling, Eds. E. Szelke and R. Kerr, North-Holland, pp. 93-106.

[21] Shoham, Y. (1993) Agent-Oriented Programming, Artificial Intelligence, N 60, pp.51-92, Elsevier.

[22] Sousa, P.; Ramos, C. (1997) A dynamic Scheduling Holon for Manufacturing Systems, Proceedings of the 2nd World Congress on Intelligent Manufacturing Processes & Systems, Buapest, Hungary, 10-13 Jun 1997.

[23] Szelke, E.; Kerr, R. (1994) "Knowledge-Based Reactive Scheduling", *Int. Journal of Production Planning & Control*, Vol 5 N 2, Taylor & Francis.

[24] Tonshoff, H.; Aurich, J.; Winkler, M. (1995) On The Way to Autonomous and Cooperative Manufacturing Systems, Proceedings 1st. World Congress on Intelligent Manufacturing Processes & Systems, pp. 420-435, Puerto Rico.