17

HOLOS : a methodology for deriving scheduling systems

Ricardo J. Rabelo [a] ; *L.M. Camarinha-Matos* [b]

[a] *scholar of CNPq - Brazilian Council for Research - at New University of Lisbon - Portugal (e-mail : kadu@uninova.pt)*

[b] *New University of Lisbon and UNINOVA - 2825 Monte da Caparica - Portugal (e-mail : cam@uninova.pt)*

Abstract

This article presents a methodology - HOLOS - for deriving particular multiagent dynamic scheduling systems from a generic architecture. An object-oriented approach is used to support information modeling and knowledge representation. The integration aspects are addressed since agents are heterogeneous and have to access external information sources. A brief explanation about the HOLOS System Generator, an interactive system deriver based on the HOLOS methodology, is also given, as well as a general description of the prototype being developed at the UNINOVA FAS/FMS pilot system. Finally, some comments are made on the current status and next steps of the work.

Keywords

Dynamic Scheduling, Multi-Agent System, Architecture Derivation, Negotiation, CIM-OSA, Integration, Virtual Manufacturing.

1 MOTIVATION AND PROBLEM DEFINITION

The current industry scenario is changing very fast due to the need of industries being competitive. After an initial period in which a simple replacement of the human resources by machines was pursued, human-centered approaches are arising as a balanced alternative in face of some negative results of total automation (Nakazawa,1994).

The dynamic scheduling activity (dynamic task assignment to production resources along the time) is directly related to resources management, which includes not only equipments but humans too. Various industrial technological waves have emerged as a result of the competitiveness requirements, and the architectures of the scheduling systems cannot stay immune from that. From an emphasis on scheduling optimality, the industry has passed to scheduling flexibility, and turning fast to scheduling agility, i.e., a scheduling system which can support an agile manufacturing towards the extended enterprise paradigm. However, even within this current dynamic scenario, industries keep demanding for custom tailored dynamic scheduling systems, which besides being open, should support a flexible (re)configuration of themselves so that they can be adapted once new production methods, scheduling control procedures, layout of the production resources, etc., are changed.

This paper introduces HOLOS : a methodology for deriving open, agile and (re)configurable dynamic scheduling system for a particular enterprise based on a Generic Architecture. The HOLOS System Generator - a computer aided derivation system - is briefly explained as well.

L. Camarinha-Matos et al. (eds.), *Balanced Automation Systems*

2 A MULTIAGENT APPROACH

The development of a dynamic scheduling system which copes with the all mentioned aspects above is still a challenge. The Multi-Agent Systems (MAS) paradigm (Huhns,1987) has arisen as a powerful approach to develop a supporting framework, specially when the following points are taken into account :

- scheduling domain is intrinsically distributed;
- scheduling domain requires a joining of different expertises;
- MAS supports a dynamic domain;
- MAS potentially supports both agents autonomy and decentralized schedules, two key factors for reaching the desired scheduling agility;
- MAS can support the integration of different problem-solving (represented in different paradigms) in the same framework.

The Generic Architecture (GA) created within the HOLOS approach - and which will be the base for a derivation - is a logical collection of distributed agents which can perform a schedule and be supervised during its execution. Applying a holistic approach in the proposed distributed scheduling architecture signifies getting a 'state of harmony' within the enterprise via a complementary and cooperative relationship between the agents involved in the scheduling (Rabelo,1994a). Such state of harmony in an agile context is a tough task to be reached. The manufacturing environment is dynamic, normally over-constrained and unpredictable to some extent. Besides that, current constraints are commonly conflicting to each other so that a trade-off or requirements relaxations have to be *negotiated.* A contract net protocol / Negotiation paradigm (Davis,1983) appears to be a suitable mechanism for supporting the desired flexibility in conflicts resolution on that holistic relationship during a schedule generation and execution. Figure 1 illustrates such a Negotiation process in our approach for scheduling. It consists of a process that leads agents to exchange information with other agents about a given business processes' requirements (Figure 1a) until a production resource agent is selected to execute it (Figure 1b).

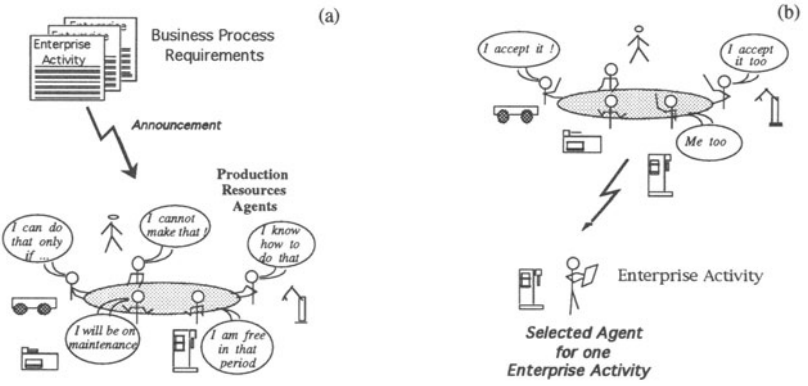


Figure 1 Negotiation in scheduling.

The general intended scenario is illustrated in the Figure 2, which can be viewed as an example of a derivation from the GA. It represents a dynamic view on the cooperative and complementary agents' behavior so that all enterprise's production structure can flexibly adapt itself to attend the arrival of a business process. In fact, this illustration joins the three vectors under which the HOLOS GA is based :

- Virtualization of the enterprise's production structure.
- Integration and information modeling.
- Multiagent distributed control.

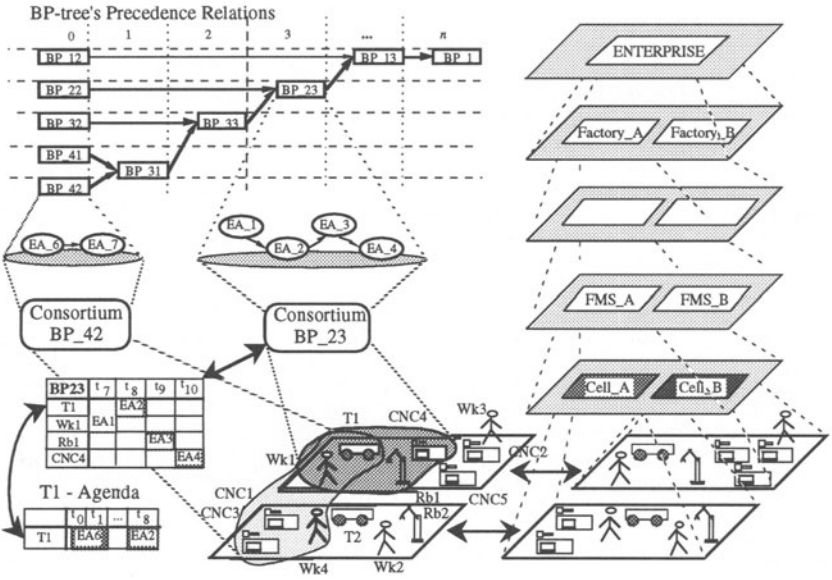


Figure 2 A Virtual Manufacturing scenario.

Scheduling cannot be seen as an isolated CIM activity. A dynamic scheduling system needs reliable and timeliness information from several sources. A global CIM Information System (CIM-IS) (Osorio,1993) containing common information models is the main vehicle through which all heterogeneous subsystems can communicate to each other, and hence the source of all information needed for the scheduling (Figure 3).

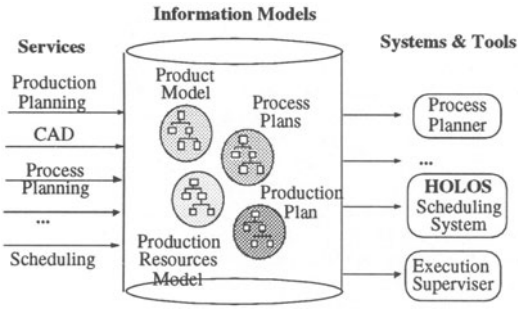


Figure 3 A CIM Information System.

Four types of agents are used in the GA. They are hierarchically related, heterogeneous, have different levels of autonomy, and have an explicit control link between each other. Briefly :

Scheduling Supervisor agent (SS)

It is a semi-hierarchical supervisor agent with the following basic functions : loading both sets of business processes (BPs) to be executed (BP-trees) and involved information models from the CIM IS; definition of business processes' requirements and their sending to the Local Spreading Centers; creation of Consortia; high level changes on planned BPs; high level actions for conflicts resolution; and visualization and scheduling evaluation.

Local Spreading Center agent (LSC)

It is a decentralized control structure for spreading the BPs announcements (requirements) through a network of production resources agents. Its main function refers to negotiate with these agents about BPs' requirements in order to select (based on some criteria) the more adequate ones for the execution of each BP.

Enterprise Activity Agent (EAA)

It is the agent responsible for executing a task itself (an enterprise activity - EA). In fact, it represents a virtualization of the production resource's local controller. In Figure 2 this agent is illustrated as human resources (workers Wk1,...,Wk4), robots (Rb1 and Rb2), etc. Its essential functions are receiving BPs/EAs requirements, their evaluation and further answer to a LSC about its temporal and technical capabilities to execute them, and a 'self-supervision' activity (in order to guarantee an EAA will only participate in a negotiation when it is 'operational'). The problem related to legacy systems is tackled in the chapter 5.1.

Consortium

It is a temporary and logical clustering of EAA dynamically selected (via negotiation) to execute a whole BP. Each Consortium has its own and local schedule, which means the global scheduling is decentralized. In the example shown in Figure 2, there is a set of business processes (BP_i) with a precedence relationship between them. BP23 for instance is composed by four enterprise activities, EA1 : EA4. Each one requires a specific type of production resource for its execution. Thus, Consortium BP23 represents the team of production resources selected to execute BP23 (T1 ,Wk1, Rb1 and CNC4). However, the Consortium BP42 needs T1 and Wk1, which in turn are also assigned to Consortium BP23. Therefore, due to the precedence relation between those two BPs, T1 has to execute EA6 before EA2. An EAA can belong to several Consortia along the time, which generates EAA contention and hence temporal constraints. Since an EAA finishes the execution of contracted EA(s) for some Consortium, this EAA becomes free both to execute other EA(s) already contracted for another Consortium and to look for more EAs, which are still waiting for execution proposals. At the end of an entire BP execution the Consortium agent kills itself.

The Consortium improves the traditional Group Technology Cell concept since it supports several types of flexibility (classified in (Chryssolouris,1992)), such as internal routing, product, volume and production. In other words, it provides the base to support a virtual manufacturing (Hitchcock,1994). Other concepts, like the 'logical cell' (AMICE,1993) and 'virtual production area' (Hamacher,1994) seem to be equivalent to the notion of Consortium. However, the basic difference is on the control flexibility, and on how it is managed in rescheduling situations. Due to the close link with the EAA agent, the Consortium is able to find a substitute EAA (via negotiation) when someone else fails.

3 INFORMATION MODELING

Information modeling and knowledge representation represents an extremely important aspect to support the mentioned MAS architecture. Presented work resorts to object oriented technology. It means that all agents and information structures are modeled as objects; i.e., by means of attributes (*slots*) and functionalities (*methods*).

Scheduling needs to have access to several information sources for / during its execution, as well as to other ones directed related to the architecture's approach. Two of essential sources for scheduling are the process plans and production resources. Their models composition have been inspired on some international projects (like IMPPACT (Gielingh,1993) and CIMPLATO (Bernhard,1992)) and on the results of the STEP community to some extent (Schenck,1994). Figure 4 shows an example of a process plan, whereas Figure 6 illustrates (in the EAA agent model) part of a production resource model.

CIM-OSA (AMICE,1993) concepts have been used for modeling the dynamic processes of an enterprise. In this sense, a production plan is modeled as a set of business processes and enterprise activities (EAs), and Procedural Rules Sets (PRSs) as the link between them. These entities form a *BP-tree* when seen as a whole (Figure 5). The negotiation process between agents is mainly based on the BP-tree's entities. However, these entities' models have been extended in order to improve the efficiency and negotiation flexibility (Rabelo,1994a).

Open solutions also include people talking a common language (terms and their precise (semantic) meaning) in order to avoid misunderstanding and, to some extent, to take the local culture into account. Some efforts in creating 'standard glossaries' have been made (Camarinha,1991). Thus, the glossary is applied on all interactive interfaces and reports.

The four types of agents used in the generic architecture are modeled as *classes* of agents, from which instances of agents are created and filled in during a derivation process. Examples of the agents classes are shown in Figure 6.

4 HOLOS - THE METHODOLOGY

The process of deriving (an instance-of) a dynamic scheduling system from a generic architecture is not 'anarchical' but based on a method. The HOLOS, a methodology to support such derivation, has been in development at UNINOVA. It corresponds to a sequence of interdependent general procedures and considerations which are to be followed by a 'human deriver' towards the implantation of a dynamic scheduling system for a particular enterprise.

CIM-OSA appears as the most prominent and wide methodology related to derivation of CIM architectures from an abstract and general reference model. HOLOS methodology is more restricted in scope than CIM-OSA, since it specifically addresses scheduling systems development as well as it is tightly biased by with the multiagent approach.

A scheduling system derivation process is normally too complex. It comprises lots of parameters and information about production, engineering and scheduling control, which in turn may be combined to each other. An open solution (the derivation) for that scenario requires an exhaustive discussion between all people engaged in, as well as the evaluation of its impacts on existing technology and human resources management. Such discussion can involve not only technical points (like production system, integration, heterogeneity of existing systems, production resources layout, solution costs, etc.), but also the enterprise's organizational culture and its work organizational methods (like team work, decentralization levels and autonomy for decision making, human resources qualifications, their functions (re)definition, training policies, etc.) as well as national singularities (like work shifts related to special holidays or local tradition, efficiency criteria specifications, etc.). The consideration of all these questions may determine the success of the system and of its implantation (Jones,1992), i.e., that the expected particular system can be achieved.

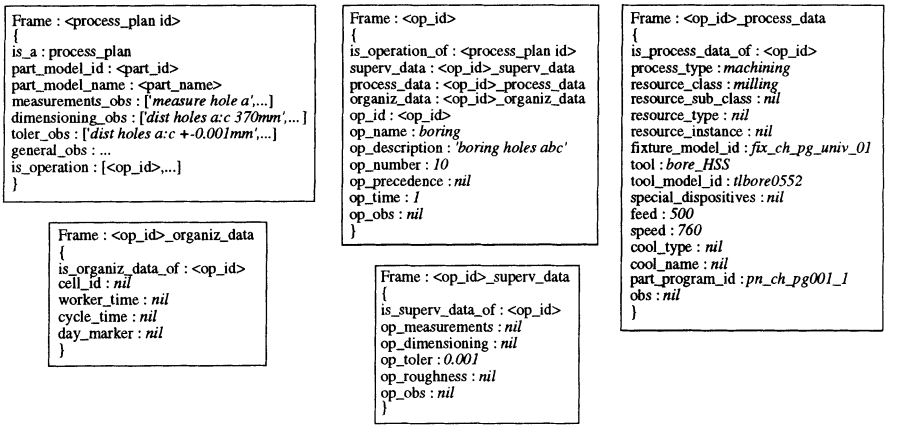


Figure 4 Example of a process plan model.

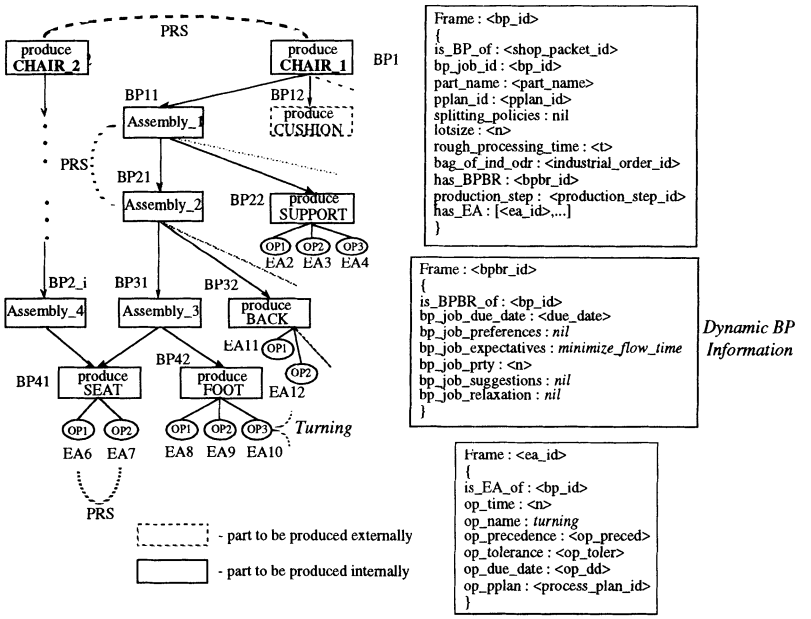
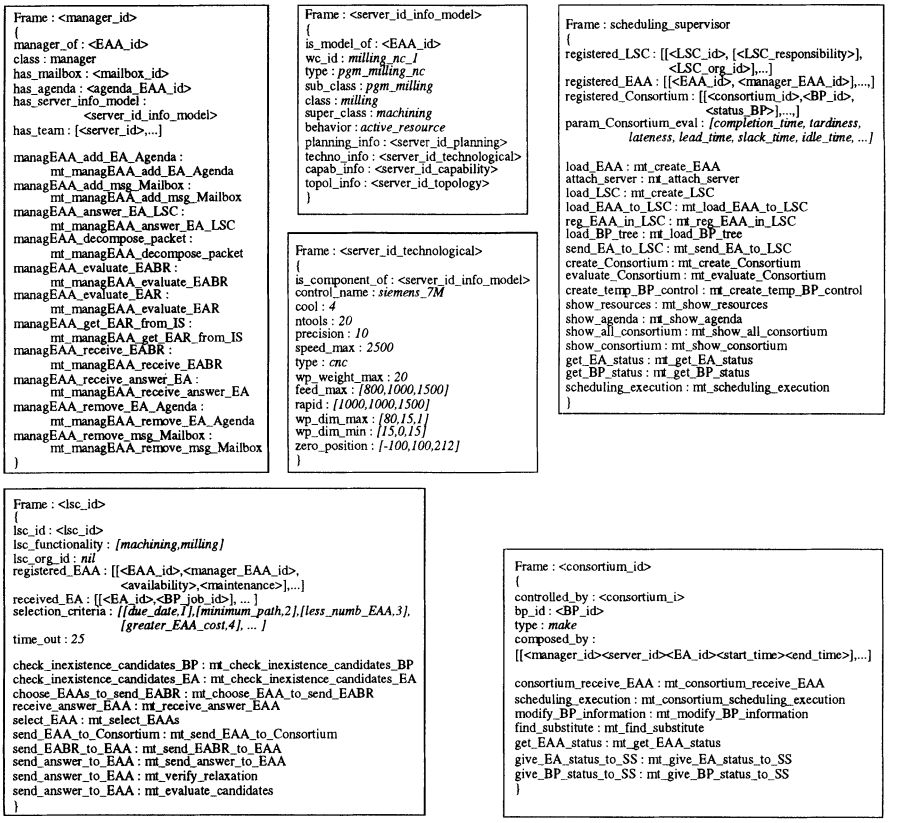


Figure 5 Example of a BP-tree model.

Although all these concerns are pointed as crucial, they are out of scope of the HOLOS methodology itself. Hence, the HOLOS methodology assumes the whole phase mentioned above is done **before** starting a derivation process. In other words, it presupposes that a

preliminary analysis and a rough system evaluation is made and their results are incorporated in the form of :

- information modeling of all entities directly related to the scheduling activity (production resources and their layout and topology, process plans and production plans);
- information modeling and knowledge representation of all entities directly related to the scheduling architecture (agents' classes and the control flow between them, glossary, communication protocol, negotiation entities and events to be treated);
- identification of the information flow between dynamic scheduling with supervision and planning actions;
- set of procedures to be used in the particular system according to the parameters and information about engineering, production and scheduling control (EPS).



EAA agent class / LSC agent class **SS agent class / Consortium (C) agent class**

Figure 6 Example of the agents' classes.

In the HOLOS approach, a derived dynamic scheduling system is represented as a particular configuration of agents organized for a concrete scenario, and that can be supervised during the execution of a scheduled production plan. A derivation basically consists in creating instances

of the agents' classes and then filling their *skeleton* along some derivation phases (discussed later). In fact, these phases represent a stepwise way through which the HOLOS methodology is utilized. Thus, hidden under those phases, the methodology's procedures are applied. Briefly, they are :

a) Agents Specification

This procedure is related to the knowledge to be incorporated into each type of HOLOS agent (SS, LSC, Consortium (C) and EAA) when instances of them are created. This knowledge is represented by attributes and functionalities, which in turn can be :

- Generic : those which each agent should have, independently of the particular site *
- Customized : the generic attributes and functionalities which need to be customized (within a set of options) for a particular scheduling system, but still independently of the particular physical site.
- Particular : the attributes and functionalities which should exist and/or have to be customized taking into account the particular physical site.

a1) Selection of EPS Criteria

The EPS aspects are directly or indirectly indicated (selected) via customized attributes. In general terms there is a *method* associated to each indicated EPS aspect. Nevertheless, the specification of an attribute may be a result of a combination of EPS aspects.

a2) Consistency Verification

The indication of the EPS aspects can be a difficult task, specially when they have to be combined to each other. A wrong specification and/or combination can provoke a situation of domain inconsistency. Two consistency verification levels exist : a simple check to guarantee that all terms indicated are defined in a glossary; and a more sophisticated analyses to guarantee a valid combination between those aspects (based on a 'derivation map', which could model all possible combinations for each aspect). Due to the complexity, this last level can suggest a decision support module may be used to help the user.

b) Agents Implantation

Implanting the agents means to make them exist in the 'world'; i.e., they can be recognized in the system, can communicate to the other agents and can execute actions. In this sense, once the logical agents instances are completed created, they do a self-announcement making use of their respective communication channels previously assigned. The 'compilation' of all agents' functionalities and other programs, the creation of libraries and adjustment of graphical interfaces are other steps to be pursued.

c) Agents Integration

In the HOLOS approach, agents have to communicate to external and heterogeneous entities in order to execute a scheduling. Such entities are the CIM-IS, other sub-systems (specially those related to the planning and supervision activities) and the production resources' local controllers. There are three possible integration layers to be made (see Figure 11) :

1- PLC (or other local controller) : Server - it aims at transforming the local production resource's controller (its PLC) in a server; i.e., creating a higher level client (in DOS or Unix for instance) which can communicate with the server. The communication process can make use of RPC, for instance.

---

* As already mentioned, the HOLOS methodology assumes that all agents' classes and information models are already composed before starting a derivation. However, a class concept can be changed (by the user), and this may be done due to some requirements of the particular site.

2- Server : EAA Manager - it aims at allowing the server to be integrated and representable into the community of agents. The paradigm client (EAA Manager / Unix) : server (Server / Dos or Unix) is applied. The communication process can make use of RPC, sockets, etc., in UDP or TCP, depending on the server's communication services.

3- EAA Manager : other agents - it aims at allowing the EAA Manager to make a conversation (in an abstract 'multiagent scheduling language') to the other agents of the architecture, to other subsystems and to the CIM-IS. A high level protocol can be utilized for that, which can be supported by RPC, sockets, etc., in UDP or TCP.

d) Agents Reconfiguration

An open solution implies giving the user the possibility to refine the Particular Architecture Infrastructure, after its generation, for the specific target system. This can be done through modifications (or extensions) and/or insertions of attributes and/or functionalities not generated during the derivation.

e) Architecture Reconfiguration

Production domain and scheduling policies and their control structures can change along the time, either due to its obsoleteness or due to some adaptation for a specific derivation (as mentioned in the Agents Specification procedure). Thus, an open architecture has to contemplate the possibility to the user for changing the agents' classes and the domain knowledge.

4.1 The Derivation Phases

Five derivation phases are utilized in the HOLOS methodology (Figure 7). Briefly they are :

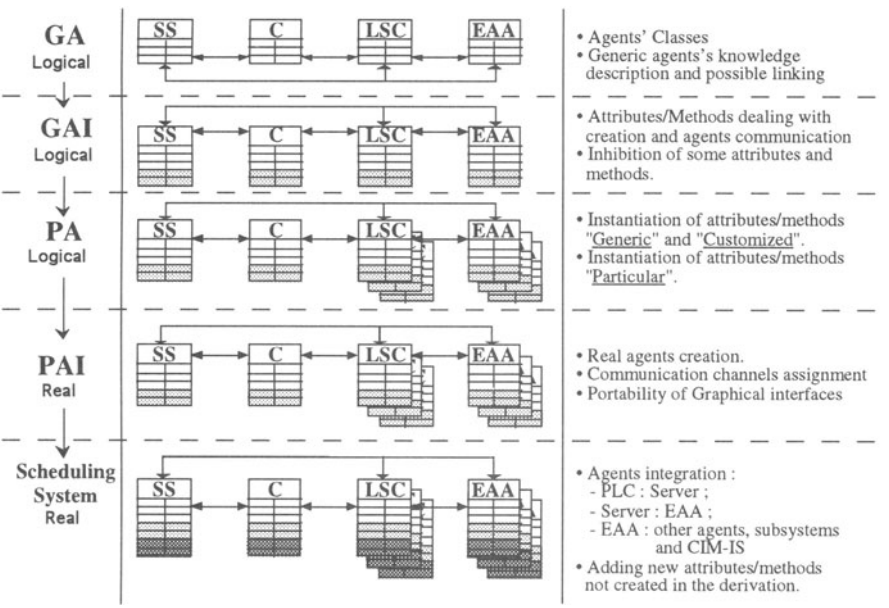


Figure 7 HOLOS derivation phases.

Generic Architecture (GA)

In this initial phase all classes of agents are abstract objects. Examples of these classes were shown in the Figure 6. They represent the 'genesis' of the scheduling system.

Generic Architecture Infrastructure (GAI)

It is the first stage of the derivation process. The main goal in this phase is to give the agents the first 'seeds of life', i.e., a set of primitives related to their creation and communication (including with the CIM-IS). The RPC protocol has been used to support the agents communication. Due to the difficulties for calling RPC services from a program (a method) written in Prolog, C programs are used as an 'intermediate binding' for that (Figure 8). Generic attributes and/or methods can be inhibited for the particular system.

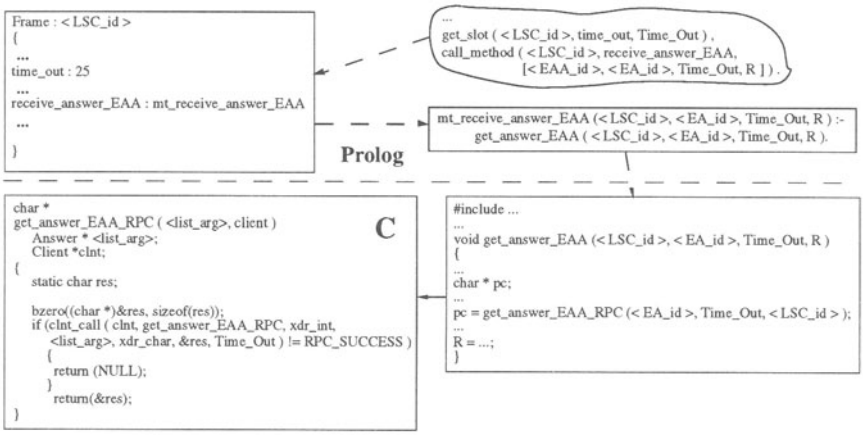


Figure 8 Example of an agent LSC in the GAI phase.

Particular Architecture (PA)

In this phase the logical instances of the agents' classes are originated. The Generic functionalities and attributes are added (by inheritance) to the respective instances, the Customized functionalities and attributes can be chosen and/or indicated from a set of options (library), and the Particular ones are specified (Figure 9). This is the first step in direction to the particular system instantiation.

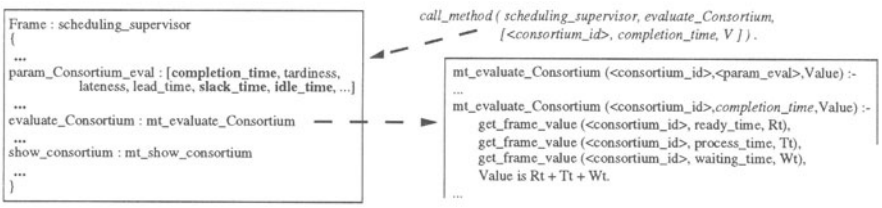


Figure 9 Example of an agent SS in the PA phase.

Particular Architecture Infrastructure (PAI)
This phase is responsible for the real creation of all logical agents instances composed in the previous phase. The communication channels have to be assigned to the agents according to their topology and enterprise model in order to connect them to each other. The adequacy of user interfaces is another aspect to take into account. In this phase, the agents become 'real' entities.

Dynamic Scheduling System
The user takes the PAI and makes the necessary adaptation for the particular system as well as handles the system (agents) implantation. An old derivation is replaced by the new one. This last phase represents a PAI completely instantiated and implanted.

5 HOLOS - THE SYSTEM GENERATOR

Likewise the other areas, the systems development technology has also suffered the 'Pendulum Law' effects. From the one extreme situation in which the systems were designed all custom-tailored and hence with high costs in development, it has passed to the other extreme in which the systems became generic, less expensive but 'black boxes'. More recently, due to the increasing of the development complexity of industrial systems and, at the same time, their need being open, modular, reusable and integratable, the 'derivation approach' has stood in significance and seems to be a balanced trend (Dietrich,1994). Its basic idea corresponds to create generic systems architectures and then to create particular 'instances-of' from that. The HOLOS System Generator (HOLOS-SG) (Rabelo,1994b) can be seen as an example in that mentioned direction. It represents an automatic way to guide a derivation. In fact, the HOLOS-SG appears to be a 'computer aided derivation' tool. By means of a strong interaction with an *user deriver* the HOLOS derivation phases are passed so that at the end of the process a PAI is generated.

It is not the objective here to describe it in details, but just to give a rough idea about it and its philosophy. Figure 10 shows its generic architecture. Some aspects deserve a brief explanation. The first one refers to the user intervention. Beyond his/her position as a deriver (and as a decision maker to some extent), he/she can alter / have access to the system concepts and libraries. Further, once the PAI is generated and in order to generate the particular scheduling system, it is necessary to implant, to integrate and, possibly, to reconfigure agents. The second one is just related to the CIM-IS role. Basically, it is the source of all information models needed for a derivation as well as the repository of a derivation representation (old ones, current one, or even one in progress). The last aspect is concerned with the rules to guide a derivation, which is supported by other structures (a Help, derivation maps and a decision support system) for consistency verification, specially in the PA phase.

5.1 Prototype under development

A test case for the NOVAFlex (Barata,1993), the UNINOVA's FMS/FAS pilot system, is in development. The objective is to derive a dynamic scheduling system based on the HOLOS methodology. NOVAFlex is composed by three robots (one *scara* and two 6 dof), two numerical control machines (a lathe and a milling), an automatic warehouse and a pallet based transport system with sensors. The current prototype has been in development and being implemented in Prolog for Aix language with an object-oriented extension (Seabra Lopes,1994), in an IBM Risc 6000 workstation.
The integration aspect is vital in manufacturing. Apart this prototype, other works on integration have been in development at UNINOVA. In short, we have faced with the legacy system problem. The production resources' controllers are quite heterogeneous, and they need to be recovered in such a way they can be integrated into the architecture infrastructure, i.e., they can be represented within the community of intelligent agents. The UNINOVA's approach is the development of encapsulating layers (as mentioned in Agents Integration in chapter 4).

The first integration layer (PLCs-Servers) is already finished for all NOVAFlex's servers, in PCs, with implementations in C and C++ languages, Linux and DOS operational systems, and X11 and TCL Tool-Kit for graphical interfaces. It means that all servers can 'offer their services' to the other agents / applications. Other works were made on how to integrate agents with other subsystems and with the CIM-IS (third integration layer). We are now concentrated in the second layer, i.e., the integration of these servers with their 'managers'. Thus, an EAA is modeled as a logical clustering of two basic interacting processes, a Manager and a Server - a tandem architecture (Figure 11). The Server, representing the resource's local controller, is a slave process which gives its Manager an allowance for offering services which it is capable to execute. The Manager 'represents' this Server within the manufacturing environment. Its basic function is 'selling' (via negotiation) the Server's services. In fact, a Manager can also represent more than one Server, depending on the production resources' topological model.

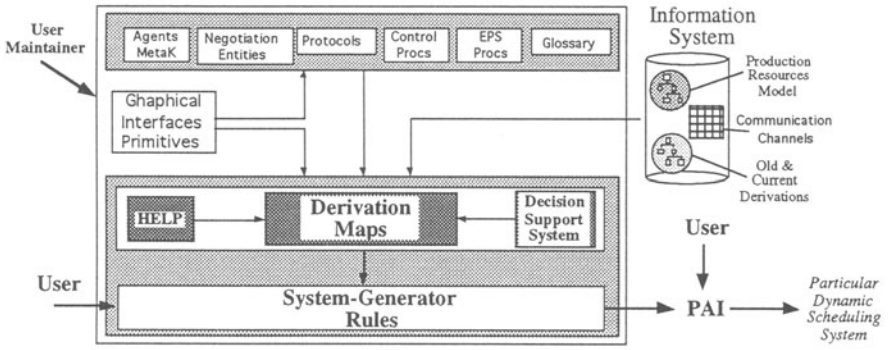


Figure 10 HOLOS System Generator architecture.

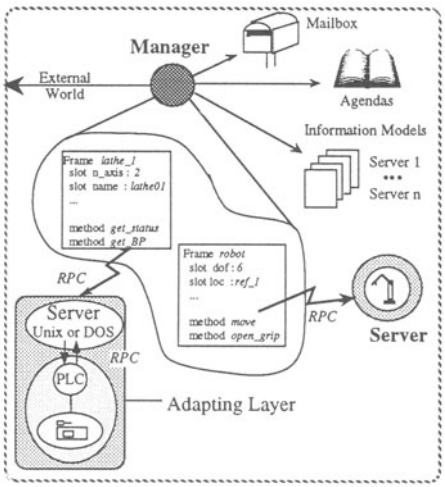


Figure 11 An example of the EAA architecture.

6 CONCLUSIONS

The HOLOS, a methodology for derivation particular dynamic scheduling systems from a generic architecture, was presented. It makes a matching with most of the characteristics considered as trends and emerging concepts in scheduling (Szelke,1994). It utilizes some anthropocentric concepts in applying a decentralized control and in exploiting the autonomy of the production resources. The negotiation between intelligent agents is used as the support technique for that.

The Engineering-Tool-Kit (Hirsh,1994) and OPIS (Smith,1994) represent two other related works. In spite of they are older, apply different approaches and are already more established than HOLOS, their general objective is equivalent : generating particular architectures and systems based on generic concepts. However, due to the HOLOS potentiality in terms of flexible modeling and control, system modularity and expandability, integration and MAS approach, a HOLOS instance system appears to fit in a more suitable way with the requirements of the industries which have envisaged for a virtual manufacturing towards the extended enterprise paradigm.

It does not intend to cover all kind of industries. The first prototype is directed to discrete and jobshop manufacturing. Further, this prototype assumes that the generic architecture is 'good enough' and that the human deriver has all knowledge on how the particular dynamic scheduling (the instance-of) has to be. However, the architecture can be modified along the time, and the instance can be adapted by the user after a derivation.

The HOLOS System Generator, a 'computer aided derivation' tool, was briefly presented, allowing the user to be assisted during a derivation.

A prototype has been in development in order to generate an instance-of for NOVAFlex. After the first implantation, the evaluation and methodology validation correspond to the main next steps to be pursued. Further, in being this work a cooperation between UNL and Federal University of Santa Catarina / Brazil, the intention is also to derive a particular system for its manufacturing cell.

7 ACKNOWLEDGMENTS

We would like to thank the UNINOVA Institute for the general infrastructure, the support provided by the JNICT CIM-CASE and ECLA Cimis.net projects, and Francisco Bernardes and Roberto Espenica for their support in the implementation of HOLOS-SG. The first author also would like to thank CNPq - Brazilian Council for Research - for the scholarship, Mafalda Leitão for her comments about HOLOS from the sociological point of view, and Gentil Lucena for his *holistic* way of being.

8 REFERENCES

AMICE (1993) CIM-OSA : Open Systems Architecture for CIM. 2nd revised and extended version, Springer-Verlag, Berlin.

Barata, J. and Camarinha-Matos, L.M. (1993) Development of a FMS/FAS System - The CRIs Pilot Unit. *Proceedings of ECLA-CIM93*, Lisbon, Portugal.

Bernhard, R., editor (1992) CIM Systems Planning Toolbox - Project Survey and Demonstration. *Proceedings of CIMPLATO Workshop on CIM Planning Tools*, University of Karlsruhe, Germany.

Camarinha-Matos, L.M., Pinheiro-Pita, H. and Moura-Pires, J. (1991) CIM Glossary - 4th Revision. UNL-Report.

Chryssolouris, G. (1992) *Manufacturing Systems : Theory and Practice*. Springer-Verlag, New York.

Davis, R. and Smith, R. (1983) Negotiation as a Metaphor for Distributed Problem Solving. *Artificial Intelligence*, **20**, 63-109.

Dietrich, B. (1994) Automation in Manufacturing, Control versus Chaos, in *Advances in Agile Manufacturing* (ed. P.T. Kidd and W. Karwowski), IOS Press.

Gielingh, W. and Suhm, A., editors (1993) IMPPACT Reference Model : An Approach to Integrated Product and Process Modelling for Discrete Parts Manufacturing. Springer-Verlag, Berlin.

Hamacher, B., Klen, A. and Hirsh, B. (1994) Production Management Elements for the Learning Enterprise. *Proceedings of IFIP WG5.7 Conference on Evaluation of Production Management Methods*, Gramado, Brazil.

Hirsh, B., Kuhlmann,T. and Marciniak, Z. (1994) Engineering Tool-Kit for Implementation of Shop Floor Control Systems. *Proceedings IFIP WG5.7 Conference on Evaluation of Production Management Methods*, Gramado, Brazil.

Hitchcock, M. (1994) Virtual Manufacturing - A Methodology for Manufacturing in a Computer. *Proceedings of Workshop on The Automated Factory of the Future : Where do we go from here ? / IEEE 1994 International Conference on Robotics and Automation*, San Diego.

Huhns, M., editor (1987) *Distributed Artificial Inteligence*. Pitman Publishing / Morgan Kaufmann Publishers, San Mateo, USA.

Jones, B. (1992) Essential Cultural Aspects, Strategies and Techniques - A Comparative View of Work Technology and Flexible Production, in *Flexible Manufacturing Systems and Work Reorganization* [in portuguese] (ed. Ilona Kovács at all), Lisbon, Portugal.

Nakazawa, H. (1994) Human Oriented Manufacturing System, in *Advances in Agile Manufacturing* (ed. P.T. Kidd and W. Karwowski), IOS Press.

Osorio , A. and Camarinha-Matos, L.M. (1993) Information based control architecture for CIM. *Proceedings IFIP Conference Towards World Class Manufacturing*, Phoenix, USA.

Rabelo, R. and Camarinha-Matos, L.M. (1994a) A Holistic Control Architecture Infrastructure for Dynamic Scheduling. *IFIP KBRS94 - Knowledge-Based Reactive Scheduling Workshop,* Budapest, Hungary. In press by Chapman & Hall.

Rabelo, R. and Camarinha-Matos, L.M. (1994b) Generation of Multi-Agent Infrastructures for Dynamic Scheduling and Control Architectures. *Proceedings 27th ISATA / Conference on Lean/Agile Manufacturing in the Automotive Industries*, Aachen, Germany.

Schenck, D. and Wilson, P. (1994) Information Modelling : The EXPRESS Way. Oxford University Press.

Seabra Lopes,L. (1994) GOLOG 2.0- A Frame Engine in Prolog.Technical Report UNL12-94.

Smith, S. (1994) Configurable Systems for Reactive Production Management, *in IFIP Transactions - Knowledge-Based Reactive Scheduling* (eds. E. Szelke and R. Kerr), North-Holland.

Szelke, E. and Kerr, R. (1994) Knowledge-Based Reactive Scheduling, *in IFIP Transactions - Knowledge-Based Reactive Scheduling* (eds. E. Szelke and R. Kerr), North-Holland.

9 BIOGRAPHY

Mr. Ricardo J. Rabelo received his degree on Computer Science in 1984, worked as consultant for several Brazilian companies as a collaborator of GRUCON / Federal University of Santa Catarina, and he is actually taking his Ph.D. at New University of Lisbon / UNINOVA on Robotics and CIM. His main interest are : dynamic scheduling and virtual manufacturing.

Dr. Luis M. Camarinha-Matos received his Computer Engineering degree and Ph.D. on Computer Science, topic Robotics and CIM, from the New University of Lisbon. Currently he is auxiliary professor (eq. associate professor) at the Electrical Engineering Department of the New University of Lisbon and leads the group of Robotics Systems and CIM of the UNINOVA's Center for Intelligent Robotics. His main research areas are : CIM systems integration, Intelligent Manufacturing Systems, and Machine Learning in Robotics.