INVITED PAPER

An Agent-Based Approach to Holonic Manufacturing Systems

K. Fischer
Department Deduction and Multi-Agent Systems, DFKI GmbH
Stuhlsatzenhausweg 3 D-66123 Saarbrücken Germany
Phone: +49-681-302-3917
Fax: +49-681-302-2235
e-mail: Klaus.Fischer@dfki.de

Abstract

This paper presents a new approach to the design of the architecture of a computer-integrated manufacturing (CIM) system. It starts with arguing why decentralised approaches to model CIM systems are superior to centralised ones and presents the basic ideas of novel approaches which are best characterised as *fractal* or *holonic* systems. The paper argues that software agents are the ideal means to implement such systems and presents the agent architecture InteRRaP for agent design. InteRRaP is then used to describe a hierarchical planning and control architecture for a CIM system, which is separated into the layer of the production planning and control system, the shop floor control systems, the autonomous system layer, and the machine control layer. Two application scenarios are described at the end of the paper. While one of these scenarios is more research-oriented, the second one is directly related to an industrial real-world setting.

Keywords
Multi-Agent Systems, Manufacturing Control, Scheduling, Optimisation

---

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35390-6_58

L. M. Camarinha-Matos et al. (eds.), *Intelligent Systems for Manufacturing*

1 INTRODUCTION

Before the idea of CIM (Computer-Integrated Manufacturing) (Scheer, 1993) made its way into practice, its original approach changed from a mainly centralistic view to a decentralised model. There are a number of reasons why a centralised approach to a CIM factory is condemned to fail (e.g., faults in individual components bring the whole system to a halt, expert knowledge is needed to run the system, the time needed for the implementation is long etc.). Warnecke (1993) adopted the metaphor of fractals to describe a model for a *flexible manufacturing system* (FMS) in which self-contained entities organise themselves without the power of an external force. In a related approach the term *holon* is used to describe an identifiable entity of a FMS which can itself be decomposed into entities of similar structure (Deen 1994b, Hasegawa et al. 1994). The term holon is a combination of the Greek word *holos*, meaning *whole* and the suffix *on* meaning *particle* or *part.* The Hungarian author and philosopher Arthur Koestler (1989) proposed it to describe a basic unit of organisation in biological and social systems. Koestler observed that in living organisms and in social organisations entirely self supporting, non-interacting entities did not exist. Every identifiable unit of organisation, such as a single cell in an animal or a family unit in society, comprises more basic units (plasma and nucleus, parents and siblings) while at the same time forming a part of a larger unit of organisation (a muscle tissue or a community). Hence, a holon is an identifiable part of a system that has a unique identity, yet is made up of sub-ordinate parts and in turn is part of a larger whole.

This paper claims that multi-agent systems are the natural means to design and implement fractal and holonic software systems. It presents an hierarchical planning and controlling structure for the design of a FMS and advocates the agent architecture InteRRaP as the basis for a FMS design according to a fractal or holonic framework.

2 THE AGENT ARCHITECTURE INTERRAP

The main idea of InteRRaP (Müller 1996) is to define an agent by a set of functional layers, linked by a communication-based control structure and a shared hierarchical knowledge base. The layers correspond to the three basic tasks an agent has to perform in an FMS: (1) co-ordination with agents on the same or on a superior layer of the FMS (*co-operative planning layer (CPL)*); (2) local problem solving (*local planning layer (LPL)*); (3) execution of the local plan (*behaviour-based layer (BBL)*) and by doing so communication with the agents of the subordinate layer of the FMS.

The CPL contains mechanisms for devising joint plans. It has access to protocols, a joint plan library, and knowledge about communication strategies. The LPL contains a planning mechanism which is able to devise local single-agent plans. The plans are non-linear data structures the nodes of which can be either new sub-plans, executable patterns of behaviour, or primitive actions. Thus, the plan-based layer may activate patterns of behaviour in order to achieve certain goals.

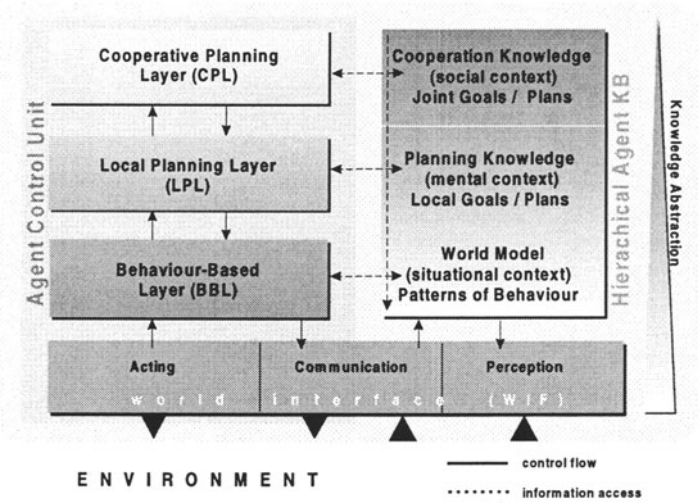

Figure 1: The Agent Architecture InteRRaP

The BBL implements the basic behaviour of the agent using behavioural scripts called *patterns of behaviour.* The BBC is closely linked to the *world interface (WIF),* and thus, to the actions and changes in the world. Patterns of behaviour can be activated both by the plan-based layer (top-down activation) and by external trigger conditions (bottom-up activation). Corresponding to the control layers of the agent, the agent's *knowledge base (KB)* consists also of three layers. The lowest layer which belongs to the BBL contains facts representing the world model of the agent. Layer two which corresponds with the LPL represents the mental model which contains local goals and local plans. Finally, layer three on the CPL specifies the social model which contains knowledge of and strategies for co-operation, e.g. beliefs about other agents' goals. The basic idea is that information is passed only from lower layers of the knowledge base to higher layers. For example, the plan-based layer can access information about the world model, whereas the BBL does not have access to planning or co-operation information.

3 HOLONIC MANUFACTURING CONTROL

The complex task of planning and controlling the manufacturing process can be distributed among various layers: production planning and control, shop floor control, autonomous system control, and machine control. Each of these layers has a clearly defined scope of competence. Splitting the act of planning and controlling into different layers brings about the advantage that the lower layers can be accessed directly without using the upper layers. Using the machine control layer, a machine can be prepared for a specific task. On the autonomous systems layer, a human operator may serve a machine tool with a workpiece. With manufacturing orders of high priority, it may be necessary to be able to pass them to the shop floor

control layer directly, without using the production planning and control layer. It is only important that the actions which access the deeper layers directly do not hamper the correct execution of the tasks in the upper layers.

Each planning and controlling entity has to handle (1) the co-ordination with the entities on the same and the next higher level, (2) the actual local problem solving (i.e. planning), and (3) the execution of the computed solution. Section 2 already introduced the agent architecture InteRRaP which supports this approach with its three-layered structure. To not confuse the layers of the agent architecture with the layers of the hierarchical planning and controlling structure, we call the latter FMS layers. However, from this trouble in naming concepts we can see that we actually do find nested structures of similar kind; just as the holonic framework suggests.

3.1 The FMS Layer: Production Planning and Control

The *production planning and control (PPC)* system is the highest controlling body to reach the goals defined by the management of a company. The registration of client orders takes place in the PPC system where they are handled according to economic criteria. Here, the classical functions: administration of client orders, data management, material management, and time management, are fulfilled. Planning of a specific client order is done with respect to the due date and the sequence of the working steps to be done. In doing so a rough estimation of the capacities of the resources needed for this client order is computed (Kupec1989). By an off-line planning step working plans which determine the manufacturing sequence for a certain end product are given to the PPC system. These working plans are passed as manufacturing orders to the shop floor control system where they are executed. All economic aspects in planning are done by the PPC system and during the design of the working steps and working plans by a product engineer. For the lower planning layers it is above all important to execute the manufacturing orders defined by the PPC system efficiently with respect to the time constraints specified by the PPC system. The more deterministic the execution of the manufacturing orders takes place, the more it is possible for the PPC system to predict when the execution of client orders will be finished.

3.2 The FMS Layer: Shop Floor Control

The PPC system passes client orders as manufacturing orders to the *shop floor control (SFC)* system. For each manufacturing order the release date and the due date is specified by the PPC system. Because all the economic aspects of a manufacturing order are considered in the PPC system, the main problem to be solved in the SFC system is how to execute the manufacturing orders efficiently and how to meet the time constraints specified by the PPC system.

In the control of the flow of material the main problem to be solved is the management of the resources and their allotment to the different manufacturing orders. In this setting, a set of manufacturing orders $O_1, \ldots, O_n, n \in N$ is given. For each manufacturing order a sequence of working steps (i.e. the working plan) $O_1 = (S_{1,1}, \ldots, S_{1,m1}), \ldots, O_n = (S_{n,1}, \ldots, S_{n,mn}), m1, \ldots, mn \in N$ is specified, which have to

be executed on a set of machines $M_1, \ldots, M_k$, $k \in \mathfrak{N}$. We use a constraint solver to compute a schedule. To deal with the computational complexity, we use heuristics to guide the search for a schedule. In a flexible manufacturing system the problem is not only to distribute a set of manufacturing orders to a set of machines. Additionally, manipulation and transportation tasks have to be planned. Because the period in time a processing, manipulation, or transportation task will last is not known exactly, a global optimal plan suffers from high uncertainties. Because of these difficulties, we use the schedule produced by the SFC system just as a predictive guideline for the local decision making done in the FMS layer of the autonomous systems. When a schedule is available the SFC controls its execution by announcing tasks for working steps which are allowed to be executed, i.e., all in the working plan preceding working steps have been successfully executed. While the execution of the schedule is going on, the constraint solver is continuously trying to find better solutions. This gives the scheduler the abilities of an anytime algorithm (Zilberstein and Russell 1993).

For the settings described in Section 4 we can assume that all autonomous systems are subordinate to one SFC system. In practice on the one hand a factory consists of several buildings which represent independent production units. In this case it is not very meaningful to control all of these production units by one SFC system. On the other hand, there might be groups of autonomous systems (flexible cells) within a production unit working closely together which gives reason to introduce a separate SFC just to control the processes within such a flexible cell. This design reflects the holonic structure of our approach. Autonomous entities (i.e. agents) form larger groups from which they can no longer be distinguished form the out-side world.

3.3 The FMS Layer: Autonomous Systems

This section illustrates in more detail how the agent architecture InteRRaP is used to design the *autonomous systems* (AS) in a flexible manufacturing system. Note that although this paper outlines only the InteRRaP agents for the AS FMS layer, the PPC and SFC entities are also designed according to this basic architecture. Recall that an InteRRaP agent is separated into three layers of abstraction: the co-operative planning layer, the local planning layer, and the behaviour-based layer. In the co-operative planning layer the ASs communicate with the superior SFC or FCC system and with other ASs on the same FMS layer. Task planning, i.e., the actual problem solving process is done in the local planning layer. Finally, the behaviour-based layer controls the actual task execution process.

3.3.1 The Co-operative Planning Layer

The SFC system passes tasks to the AS FMS layer as soon as it is determined by the production plan that a task may be executed because all of the in the working plan preceding working steps have been successfully completed. The SFC system does not care if it is possible for a group of ASs to execute this task immediately or if they are currently engaged in the execution of a task. The SFC system just inserts

the task into a list which is accessible to all involved ASs and the ASs decide by themselves when it will actually be executed. By doing a specific task several ASs have to co-operate. Each AS has to play a part to solve a specific task. No AS may believe that it is the only one which wants to play a certain part for a specific task. Therefore, the ASs must coordinate their intentions of playing parts in different tasks on the *co-operative planning layer (CPL)*. The main problem to be solved on the CPL is the problem of finding a consistent group of ASs which all together are able to solve a specific task. We call such a group a *complete team* for a task. Only tasks for which a complete team was found can actually be executed.

The ASs are separated into three groups: **R** mobile manipulation systems (mobile robots), **T** transport systems, and **M** machining centres and locally fixed robots. In some setting even the workpieces which are to be processed are able to move autonomously, for example when they are installed on a transportation device. In these settings it is meaningful to control these workpieces by autonomous agents. We therefore introduce the set of workpieces **W**.

Mobile manipulation systems are able to work flexibly on the given tasks. Each time a mobile manipulation system finishes the execution of a task it can start to solve all the tasks it is able to regardless of its current configuration. For transport systems each transportation task may be split up into the subtasks: load, transport, and unload. It is possible to give to a transport system the transportation task in the whole or to give the three subtasks to the transport system separately. When there is only one task specified for the whole transportation task, all the ASs involved in this task, the one to load and unload the transport system, are bound when the execution of the transportation task is started. If loading, unloading, or the transport task itself needs much time these systems might be bound unnecessarily long.

Locally fixed robots, machining centers; and flexible cells are much more restricted in their ability to choose tasks to be executed than ASs in $\mathbf{R} \cup \mathbf{T}$ because ASs in **M** are fixed to a location. An AS a in **M** depends on ASs of $\mathbf{R} \cup \mathbf{T}$ if not all of the devices needed for a specific task are already present within a. We therefore introduce the precedence relations $\mathbf{W} < \mathbf{M} < \mathbf{T} < \mathbf{R}$. $\mathbf{T} < \mathbf{R}$ means that a member of **R** may only join a team for a specific task if all of the members of set **T** already joined the team for this specific task. The precedence relation $<$ is transitive that means that for example $\mathbf{W} < \mathbf{R}$ is valid too. The idea behind this definition is that the ASs which are able to execute tasks flexibly may react to the decisions of ASs which lack this flexibility in task execution.

To find complete teams, the ASs examine the list of tasks, which are announced by the SFC system, and try to reserve the task they would like to execute next for themselves. When an AS was able to reserve a task successfully for itself, this AS becomes the leader of the team for this task. The leader l of a team for a task t has to find a complete team for this task. l does this by sending messages to the other ASs which ask these ASs to join the team for task t. A conflict occurs if two team leaders send each other messages in which each of them asks the other one to join its own team. It is possible to describe conflict resolution protocols for this situation which guarantee lifeness and fairness of the system.

3.3.2 *The Local Planning Layer*

The CPL of an AS passes a local task description to the *local planning layer (LPL)* of the AS which has to be solved in order to solve the global task given on the CPL. By solving a task the ASs have to co-operate with each other. For that reason, the complex task given by the CPL has to be decomposed into primitive actions each of which can be executed by the AS without interacting with other ASs. The task planner contained in the LPL of an AS therefore implements the function: f: $\mathbf{K}^* \times \mathbf{T} \rightarrow \mathbf{P}^*$ where $\mathbf{K}$ is the set of a all possible knowledge base states, $\mathbf{T}$ is the set of all tasks, and $\mathbf{P}$ is the set of primitive actions. The input to the task planner is a sequence of knowledge base states because the content of the agent's knowledge base changes while task planning is active. Hence, the task planner reacts to changes in the environment of the AS by reacting to the changes in the agent's knowledge base. For each task function f is specified by a skeleton plan. Event calculus reasoning is used to adapt these skeleton plans to the current situation.

3.3.3 *The Behaviour-Based Layer*

The primitive actions derived in the LPL of an AS are executed in the *behaviour-based layer (BBL)* of the AS. By executing primitive actions, an AS does not explicitly interact with other ASs. Nevertheless, such a primitive action itself may be a complex task, for example the sensor-guided insertion of a pin into a hole by a robot. The solution to such a task may be specified with the help of sensor/actor networks (Connel 1990, Brooks 1990). Simple control commands are sent from the BBL to the machine control FMS layer where the actions are actually executed. The separation of the AS FMS layer and the machine control FMS layer is done due to the systems which are available today. The controllers of these robots and machine tools provide only limited computational power and are therefore not able to execute complex programs. In the future it is likely that the controlling units of such systems will offer the computational power of today's workstations and therefore the AS FMS layer and the machine control FMS layer will grow together.

4 APPLICATION SCENARIOS

The hierarchical planning and controlling structure described in this paper has been applied to the design of CIM systems for two application scenarios.

The first application scenario is a model (size 140x120 cm^2) of a flexible manufacturing system (see Fig. 2 (a)). Two robots serve machine tools when performing pick-and-place tasks. $Depot_1$ (lower right corner of Fig. 2 (a)) can be handled only by $robot_1$; on the other side, the three machine tools can only be served by $robot_2$. The working spaces of the two robots overlap in the area of the input conveyor belt (in the lower left corner of Fig. 2 (a)), the $depot_2$, the heating cell and the output conveyor belt (hidden by the heating cell and $robot_2$ in Fig. 2 (a)). Therefore, it is necessary to give the robots strategies to avoid collision. When serving the machine tools and loading the heating cell, the movements of the robots

have to be coordinated with the activities of the machines. The workpieces enter the manufacturing plant over the input conveyor belt, where they are identified by a barcode decoder. With the help of inductive sensors, which are attached to the conveyor belts as well as to the workpiece spots of the machines, it is possible to check the presence of a workpiece. $Depot_1$ takes up workpieces which cannot be worked on at the moment. $Depot_2$ is used to hand over workpieces from $robot_1$ to $robot_2$. The implementation of the planning and controlling hierarchy for this model was a complex task. To have a comfortable environment to develop and test our implementation we implemented a 3D simulation environment using the universal simulation system IGRIP (1992).

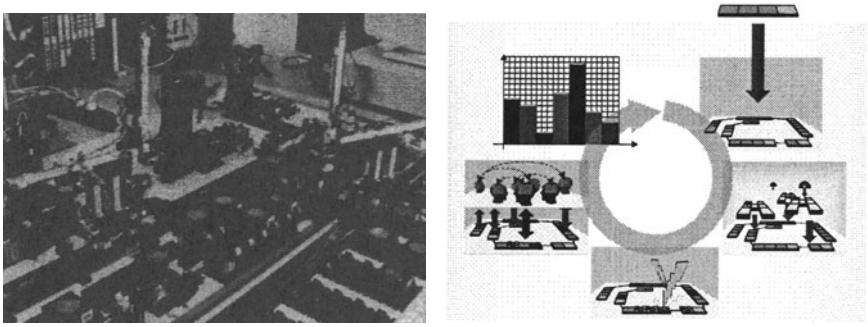

Figure 2: (a) Model of a Flexible Manufacturing Plant (b) Optimisation Cycle for Topologies

In the second scenario we have a production line which consists of a set of production units each of these production units provides one or more (in our case up to 7) processing operations. These processing operations themselves consist of one mechanised processing step and several (up to 5) manual processing steps. We call a setting with a fixed number of production units a topology. A topology specifies a concrete geometrical position and orientation for each production unit and defines the paths which workpieces can take to pass from one production unit to the next one. In practice it turns out that a topology where we have a linear sequence of production units each of which provides exactly one processing operation is most widely used. However, this topology has sever drawbacks when we take machine faults into account. Therefore we designed the control architecture in a generic manner such that it is possible that it can control any desired topology. This brought about the possibility that we can now evaluate the performance of different topologies in a simulation environment. The idea is now to run a search procedure in which different topologies (generated by using techniques genetic algorithms) are evaluated and the one with the highest performance is selected to be tested in a real-world setting (see Fig. 2 (b)).

5 CONCLUSION

This paper outlined an holonic approach to the design of a flexible manufacturing system (FMS). The starting point for doing so was a hierarchical planning and

controlling structure which consists of the layer of the production planning and control (PPC) system, the layer of the shop floor control (SFC) system, the autonomous systems (AS) FMS layer, and the machine control layer. The paper assumes a traditional PPC system and concentrates on the description of the remaining FMS layers starting from the FMS layer of the SFC system. The agent architecture InteRRaP is introduced as a basic means to describe the autonomous entities in the planning and controlling hierarchy. InteRRaP itself structures an agent into three layers of abstraction: co-operative planning layer (CPL), local planning layer (LPL), and behaviour-based layer (BBL). An agent uses its CPL to co-ordinate its activities with the agents on the same and the superior FMS layers. The LPL is used for local problem solving, i.e. to devise plans for the tasks the agent is to execute. It is the BBL which starts to execute these local plans at an appropriate point in time. The main objective of this paper was to present the overall structure of the holonic FMS design. Therefore, the local problem solving concepts of the agents were not presented in great detail. However, we investigate constraint-based planning techniques to describe the agent's reasoning, which give us a unique framework to describe and solve scheduling problems and task planning. The picture is as follows: the LPL actually solves the agent's tasks, the CPL negotiates with other agents about what the agent has found out and the BBL actually executes the solution. It is this uniform view to distributed problem solving which makes our approach extremely powerful.

6 REFERENCES

Brooks R.A. (1990) A Robust Layered Control System for a Mobile Robot. In: Winston, P. H. and Shellard, S. A. (ed.) (1990), 3–27.

Connell, J.H. (1990) Minimalist Mobile Robotics. *Perspectives in Artificial Intelligence*, **5**, Academic Press, Inc.

Deen, S.M. (1994a) Proc. of the 2nd Intl. Working Conf. on Cooperating Knowledge-based Systems (CKBS'94) (Selected Papers). Keele University, DAKE Centre.

Deen, S.M. (1994b) A cooperation framework for holonic interactions in manufacturing. In: Deen (1994a) 103–124.

Fischer, K. (1993) The Rule-based Multi-Agent System MAGSY. In Proceedings of the CKBS'92 Workshop. Keele University, DAKE Centre.

Fischer, K. (1994) The Design of an Intelligent Manufacturing System. In Deen (1994a), 83–99.

Fischer, K. (1994) Knowledge-based Reactive Scheduling in a Flexible Manufacturing System. In Kerr and Szelke (1994), 1–18.

Hasegawa, T. et al. (1994) Holonic planning and scheduling architecture for manufacturing. In Deen, S. M. (1994a), 125–139.

IGRIP (1992) IGRIP User Manual and Tutorials. Deneb Robotics Inc., Auburn Hills, USA.

Kerr, R.M. and Szelke, E. (ed.) (1994) Proc. of the IFIP TC5/WG5.7 Workshop on Knowledge-Based Reactive Scheduling. Elsevier Science B.V.

Koestler, A. (1989) The Ghost in the Machine. Arkana Books.

Kowalski, R. and Sergot, M. (1986) A Logic-Based Calculus of Events. New Generation Computing, **4**(1),67–95.

Kupec, T. (1989) Integration of Autonomous Mobile Robots in Flexible Manufacturing Systems. In *Proc. of the 2nd Inter. Conf. on Intelligent Autonomous Systems*, Amsterdam, 122–133.

Müller, J.P. (1996) The Design of Intelligent Agents—A Layered Approach. *LNAI*, **1177**, Springer.

Scheer, A.-W. (1993) CIM Computer-Integrated Manufacturing — Towards the Factory of the Future. Springer.

Warnecke, H.-J. and Hüser, M. (1995) The Fractal Company A — Revolution in Corporate Culture. Springer-Verlag.

Winston, P. H. and Shellard, S. A. (ed.) (1990) Artificial Intelligence at MIT, Expanding Frontiers, MIT Press, Cambridge, Massachusets.

Zilberstein, S. and Russell, S. J. (1993) Anytime Sensing, Planning and Action: A Practical Model for Robot Control. In *Proc. of the 13th Int. Joint Conf. on Artificial Intelligence (IJCAI-93)*, 1402—1407.

7 BIOGRAPHY

Klaus Fischer studied computer science at the Technische Universität (TU) in München. 1986–91 he worked in a joint research project SFB 331 *Information Processing in Autonomous Mobile Robot Systems* at the Department of Computer Science at the TU München. 1992 he finished his doctoral degree with his thesis on Distributed and *Co-operative Planning in a Flexible Manufacturing System.* 1992 he joined the *Multi-Agent System* Research Group at DFKI GmbH in Saarbrücken in the department of *Deduction and Multi-Agent Systems* headed by Prof. H. J. Siekmann and assumed the responsibility of group leader in November 1993.