Hybrid Control Architecture Supported By T-Temporal Petri Nets

S.I. CARAMIHAI and Adrian CURAJ

Politehnica University Bucharest, Human Resource Training Center
132 Calea Grivitei, Sector 5, Cod 78122, Bucharest, Romania
Tel. +4013111661, Fax.+4013111794, E-mail: sic@ics.pub.ro, adi@cncsis.ro

Abstract: *The paper presents hybrid agent based architecture for the control of flexible manufacturing systems. The main goal of the architecture is to solve a class of problems raised by agent based control systems: the non-optimality of the control policy, especially from the time point of view. A supervisory level is designed for this purpose, having as its main task to evaluate different possible control policies and to advise agents in choosing the optimal one. The modeling support used for this purpose is T-temporal Petri Nets.*

Keywords*: hybrid architecture, autonomous agents.*

1. INTRODUCTION

Real-time control of flexible manufacturing systems (FMS) represents one of the most complex classes of problems. Reasons are either general for large-scale systems (the amount of data to be processed with tight time constraints, heterogeneity of information, the necessity to use simplified models) or specific for FMS (frequent changes in system's structure, uncontrollable events occurrence, existence of multiple choices, conflicting evaluation criteria etc.). Basically, there are two evaluation criteria for the control architectures, that the majority design approaches proved as contradictory:

- Robustness (the capability of the control structure to bypass the effects of eventual break-downs in the controlled system so as to accomplish manufacturing goals– usually achieved by on-line rescheduling)
- Global optimality of the control policy in terms of system efficiency (one of the most important optimality criteria being the time spent by parts in the manufacturing system).

Consequently, there are two design approaches for control architectures: hierarchical and agent based.

In the case of hierarchical control architectures, the control problem is vertically stepwise decomposed at enterprise/ shop floor/ cell levels, where the basic criterion for differentiating levels is complexity (of data&process structure). Every level is furthermore decomposed in modules, based on their respective functionality, thus resulting hierarchical distributed control structures.

These structures are usually very efficient from the point of view of the global optimality of the control policy. Other advantages are the clear modularization of the control system, implying the reusability of codes, lent degradation of performances in case of breakdowns, different frequencies of processing on different levels.

From the robustness point of view, performances of hierarchical distributed architectures are not very good: usually is necessary to prepare special procedures to treat breakdowns and it proves to be extremely complicated and costly, due to the complexity of the overall system.

The other design approach is intended to ensure the robustness of the control system vs. breakdowns by the dynamic reconfiguration of the control system in answer to the reconfiguration of the physical one. This is the agent-based control approach.

The basic idea is to use a functional horizontal decomposition of the control problem in order to create specialized modules, capable to locally solve, by dedicated algorithms, pre-defined goals. The main difference between these modules and those composing static hierarchical architectures is their position versus the environment. Without an hierarchical stratification, all the basic control functions - as planning, scheduling, monitoring, data upgrading and processing - need to be performed by the same module – the agent -, based on information obtained from other similar control entities. The usual predefined goals for a control module are with regard with either the management of a resource or that of a product realization.

Several heterarchical architectures (Lin and Solberg, 1994), (McFarlane, 1995), (Wyns *et.al*., 1999) were developed for manufacturing control, having important advantages as the robustness to break downs and the facile extension of the control architecture with new modules but also important disadvantages of non-optimality and blocking.

Both disadvantages are due to the evolution of control agents in weakly defined environments and to the lack of a global view of the manufacturing system.

The first cause was partially eliminated by conceiving control agents as modules (Albus *et.al*., 1999) capable to improve their knowledge about the environment and to consequently on-line modify their internal model. The second one necessitates a superior control level, capable to evaluate the possible evolutions of the agents and to advise them – a supervisor.

The present paper proposes a hybrid agent based supervised architecture. The capability of the supervision level to evaluate the possible evolutions of agents is based on a modeling support mixing an event driven representation with continuous time integration – T-temporal Petri Nets.

2. THE INTELLIGENT SUPERVISED CONTROL ARCHITECTURE

2.1. Basic principle

The control architecture proposed in this paper consists – at the agent level - in two types of basic modules (product and resource), capable to interact with their pairs by a system of questions-answers.

The generic structure of an agent includes four basic blocks (Figure 1.):

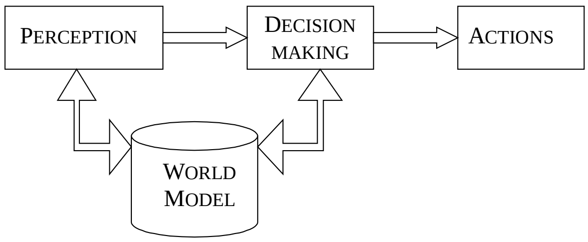

Figure 1. Generic structure of an agent.

- **The perception block** - receives every information from the environment and modifies, if necessary, the internal model of the agent
- **The decision making block** – it decides either how to process the received data (with direct and explicit consequences on the immediate actions of the control agent) or how to modify information and knowledge contained in the world model block (with implicit consequences on the whole behavior of the agent)
- **The action block** - is basically a communication interface with the environment: it either starts tasks for the controlled process or sends typified questions for other agents.
- **The World Model block** - is a knowledge base containing the internal model of the agent (including relevant knowledge about the environment, i.e. the state of temporary links with other agents, the state of the controlled processes) and a rule base on "how to take decisions".

The world model has to have intrinsic updating mechanisms for its two main parts, verifying and ensuring the global consistency of data.

2.2. Basic types of agents

As previously mentioned, the presented control architecture implies two basic types of agents, every one having a specific internal organization. These basic types are "product agent" and "resource agent".

The product agents are in charge with the realization of a specific product or of a lot of similar products by negotiating with respective resource agents.

The resource agents are controlling a specific resource in order to ensure processing for products.

The communication between different types of agents is a negotiation procedure called “centered on the product” (i.e. product agents start negotiations and, if necessary, decide about the resource agents they should choose).

The world model of a resource agent should contain:

- the list of processing tasks that the resource could perform - with their respective duration and cost
- the (up-datable) current state of the resource
- the (up-datable) history of negotiations with resource agents and their basic characteristics (solicited operations - cost requested - result of transactions)
- the (up-datable) efficiency (evaluation) function value
- information about agents with which are established in-course negotiations: solicited operations, cost, allocated priorities.

The decision making block should contain basically the rules for constructing an evaluation function for process-agents offers. These rules should be subject of modifications, based on the history of successful/unsuccessful negotiations and of the current value of the efficiency function. Other rules included in the decision making part should be with regard to the realization of physical processing, allocation of auxiliary materials a.s.o.

As concerning a product agent, its world model could be more complex, including:

- all the technological available variants of the physical basic product
- (up-datable) limits of cost functions allowed for different operations
- manufacturing constraints as the due date, priority, maximal fabrication costs, etc.
- the actual state
- the current value of its cost function and its history
- the upper time limit for resting in the manufacturing system
- the details of the in-course negotiations.

Negotiations should develop following a determined protocol and containing a typed set of questions-answers, both in order to simplify the communication procedure and mainly in order to finish in a given time limit.

In order to successfully finish a negotiation is important a mutual agreement of both parts, meaning that before allocating a resource to a product, both parts should cancel other negotiations.

This condition implies that an agent could negotiate in the same time with several others, this approach being intended to optimize the efficiency of the system in terms of cost functions.

In this case, imposing a maximum duration for a negotiation process of an agent could prove a good measure against the possibility to loose time trying to obtain the better resource, especially when other alternatives are available.

2.3. Supervisory level

The control architecture based on intelligent autonomous agents presented above has a high degree of robustness at resource break-downs because in this situation the corresponding resource agent simply doesn't answers at requests regarding the operations it cannot perform. In the same time, it is very easy and natural to introduce new agents in the architecture.

But despite the implicit robustness and adaptivity of the structure, there could appear some important problems, as:

- a critical resource is requested by two products with comparable cost functions for the operations in cause - blocking situation
- a product agent lasts in the system more than its upper time limit – delay situation

The supervisory module should be designed for solving these possible problems and, more important, in order to ensure a global optimality in the functioning of the system.

The fact that a resource doesn't treat a request immediately and waits for a pre-defined (and eventually adjustable) period of time in order to receive several, if possible, doesn't improve seriously the performance of the manufacturing system. It is necessary to have a module capable to evaluate some important performances (as duration and non-blocking) of several possible scenarios, in order to choose an optimal one.

In the present case, the supervised control architecture is designed to have the following properties:

- autonomy - at agent level and at sectorial level
- capacity of collaboration
- rapid adaptation to environmental changes.

A supervisory module is directing a group of agents. Considering their respective internal models – goals included – the supervisor is intended to combine them into a global model and to off-line synthesize the possible control policies. Finally it will select the optimal one and will decompose it in partial control policies that will be sent to product agents.

Normally, in their on-line functioning, the product agents will try to follow these “optimal” control policies. If, by different reasons, they will fail, they are free to try other negotiations, preserving a sub-optimal functionality of the system.

3. MODELING SUPPORT

One of the most important factors that guarantees the consistency and speed of all control/ supervising activities is the modeling support of the controlled system and of its goals.

As concerning the agents, their structure is designed so as the following aspects are distinct:

- the environment - represented dually by its model and its perception as feedback for actions
- the goals - decomposable in tasks
- the methods for executing tasks and for decomposing goals
- the actions on the environment as the results of tasks.

Based on these considerations it results that the world model would have a dual representation: in terms of rules for goals, methods and actions and in a dedicated formal support for the environment.

This modeling support should meet some minimal conditions as:

- to reflect the typical behavior of products/ resources from real-time control point of view
- to permit modeling at different complexity levels agents and their goals
- to allow a modular approach
- to integrate time,
- to allow condition/ rules driven evolution of models but also quantitative evaluation.

A formalism meeting these requirements was introduced in [5] as T-temporal Petri Nets. The off-line synthesis of an optimal policy by a supervisor is based on the principles presented both in (Caramihai, Alla, 1998) and (Caramihai *et al*., 1998).

An off-line synthesis will have the following steps:

1. Selection of the interesting product models - from the product agents world model (implying decisions if the products have different variants of manufacturing)
2. Selection of the necessary resource models - taking into account only the

operations requested by the actual products
3. Construction of the closed loop functioning model, based on the synchronous composition
4. Evaluation and synthesis of the control policy.

In the case of the success of step 4, the control policy obtained will be implemented as recommendation of scheduling in the decision-making blocks of involved agents.

In case of failure - go to step 1 - until no other variant of decision can be made or time allowed for off-line processing expires.

4. CASE STUDY

4.1 System description

The considered case study consists into a flexible-manufacturing cell (Figure 2) with 4 working posts, interconnected by a circular conveyor that transports parts placed on pallets.

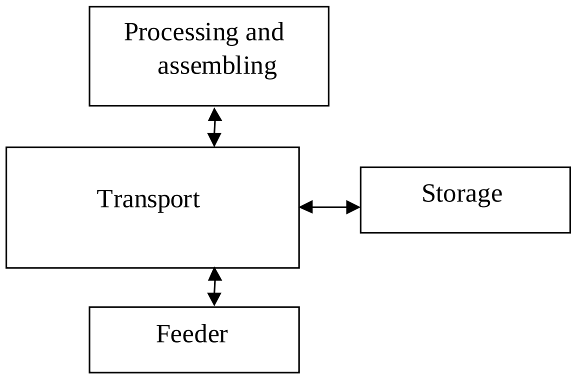

Figure 2. Flexible manufacturing cell structure.

The role of feeder station is to provide the system with raw material needed for final product manufacturing.

This workstation is composed from four types of buffers: B_1 – A type parts buffer (unprocessed cylinders), B_2 – B type parts buffer (preprocessed cylinders), B_o – cube parts buffer, B_p – pallets buffer.

Two manipulators are deposing parts on pallets and respectively pallets on conveyor.

The storage station role is to stow the pallets from conveyer. The pallets that are arriving at the storage station in order to be stowed are taken by a manipulator and deposed on one of the two available buffers until is possible either to send them to the conveyer (using a third manipulator) or to the storage (using storage station arm). The pallet repository is a 4 line by 8 columns matrix of locations, every location being used for stowing a pallet and its content. In order to stow pallets on different locations a mobile arm is used.

The transport station is composed from a conveyer and three stoppage places (one for every workstation in the system).

Processing and assembly station allows taking of one cylinder (A or B type) at the time from the conveyer, using a robot that places them to the CNC numerical control lathe. The same robot can depose cube parts from conveyer and processed parts from the lathe to the assemble station.

4.2 Functional description of the system

Accordingly to the proposed system architecture, each of the four workstations of the flexible-manufacturing cell is defined as a resource agent.

In principle, for every class of resource type agents, the supervisor has a list of agents. During the on-line functioning there are three ways of finishing an assigned task:
(a) successfully (by the agent and at the time previewed);
(b) unsuccessfully (an malfunction occurs somewhere in the operation evolution) – but the supervisor is still monitoring the functioning of the system
(c) unsuccessfully, but the supervisor does not receive any event from the implied agents – if a certain critical time for the operation in course expires and no message is received, the supervisor assumes that the agent was not capable to finish the job and declares the operation unfinished.

In b) and c) situation, the functioning of the system will continue by negotiations between agents.

An example of the respective models is presented in Figure 3:

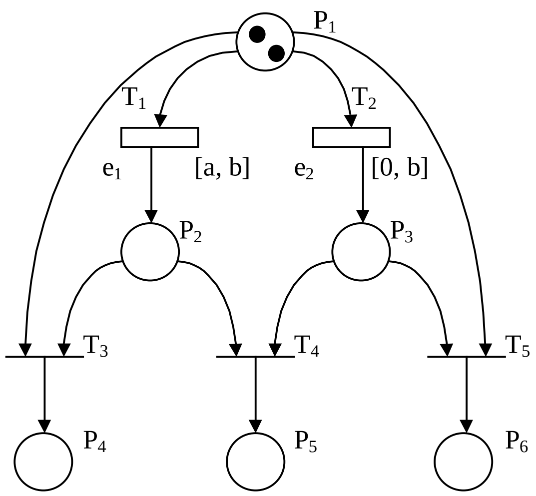

Figure 3. T-temporal Petri Net model.

The given example represents the T-temporal Petri Net model of a task. The main characteristic of a T-temporal Petri Net is that every transition T_i has assigned an event and a time interval denoted [a, b]. A token from each input place of a transition T_i is taken and a token placed in each of its output place either if the event e_j occurs in the time interval [a,b], or if the elapsed time is superior to b. The time is counted from the moment of the validation of T_i.

The place P_1 indicates the start of processing. The event e_1 represents the completion of operation. The time interval [a, b] represents the minimum and maximum time needed for completion of the operation. The e_2 event represents the impossibility to fulfill the operation (due to a malfunction). This event can occur from 0 moment of time until maximum moment of time – b.

If in the time interval [a, b] the event e_1 occurs, it means that the operation is completed. One token from P_1 is taken and one token is placed in P_2. Because of the fact that in P_1 and P_2 is the necessary number of tokens, the transition T_3 is realized and one token is taken from places P_1 and P_2 and one token is placed in P_4. So, place P_4 indicates the completion of operation (situation a)).

If the event e_2 occurs in the interval [0, b] this means that a malfunction occurs before operation completion, so the operation cannot be finished. In this case, a token from P_1 is taken and one is placed in P_3. Because of the fact that the necessary number of tokens required by T_5 transition is available, T_5 is executed, so that a token is taken both from P_1 and P_3 and a token is placed in P_6. The marking in P_6 indicates that the malfunction occurs. (the c) situation).

If the critical time for the completion of the task expires –no event occurred in the interval [a, b], it means that two tokens are taken from P_1 and a token is placed in P_2 and one in P_3. So, the T_4 transition is valid and executes. A token is taken from P_2 and P_3 and a token is placed in P_5. In this way, we are using the place P_5 to indicate that the critical time expired.

Using this strategy, a global model for the flexible-manufacturing cell can be created by the synchronous composition of respective agents' models.

The supervisor needs to be informed only on the occurrence of some events in agents functioning so as the supervisory internal model will have a dimension inferior to the net representing the whole process.

5. CONCLUSIONS

The paper presents hybrid control architecture for large-scale flexible manufacturing systems that intends to combine the advantages of agent based architectures (robustness) with those of hierarchical distributed architectures (optimization).

An optimal control policy is off-line synthesized using T-temporal Petri Nets models of resource agents, synchronous composed by the models of product agents. This policy is realized on-line, under the monitorization of the supervisor, unless certain malfunction intervenes. In this situation, agents became completely autonomous and act only considering their local optimality criteria.

REFERENCES

1. Albus,J.S., Barbara,A.J., Nagel,R.N. (1995) "Theory and practice of hierarchical control", *Proc. IEEE Computer Soc.Int.Conf. Compcon*

Falls, National Bureau of Standards, Washington DC.

2. Caramihai,S.I., Alla,H., (1998), "On the synthesis of a controllable supervisor for Discrete Event Processes modeled by Temporal Petri Nets", *Proceedings of the IEEE Conf. on Robotics and Automation - ICRA' 98*, Bruxelles, pp.285-291.
3. Caramihai,S.I., Dumitrache,I., Stanescu,A.M, (1998) "Real-Time Supervision for Intelligent Manufacturing Supported by T-temporal Petri Net Models" *Proceedings of the IEEE Conf. on Systems, Man and Cybernetics*, San Diego, pp. 582-587.
4. Lin,Y., Solberg,J. (1994) "Autonomous control for open manufacturing systems", *Computer Control of Flexible Manufacturing Systems*. (Joshi,S. and Smith,J. (Ed.)), Chapman & Hall, McFarlane,D. (1995) "Holonic manufacturing systems in continuous processing: concept and control requirements", *Proceedings of the Annual Conference of ICIMS-NOE*, ASI '95, pp. 273-282, Cascais, Portugal, June 25-28.
5. Wyns,J., Van Brussel,H., Valckenaers,P. (1999) "Design pattern for deadlock handling in holonic manufacturing systems", *Production Planning and Control*, Vol. 10, No. 7, pp. 616-626.